M 1

```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRRR           TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRRR           TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRRR           TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**BASUDFRL

```
BBBBBBBB      AAAAAA      SSSSSSSS  UU      UU  DDDDDDDD    FFFFFFFFFF  RRRRRRRR    LL
BBBBBBBB      AAAAAA      SSSSSSSS  UU      UU  DDDDDDDD    FFFFFFFFFF  RRRRRRRR    LL
BB      BB   AA      AA  SS        UU      UU  DD      DD  FF          RR      RR  LL
BB      BB   AA      AA  SS        UU      UU  DD      DD  FF          RR      RR  LL
BB      BB   AA      AA  SS        UU      UU  DD      DD  FF          RR      RR  LL
BB      BB   AA      AA  SS        UU      UU  DD      DD  FF          RR      RR  LL
BBBBBBBB     AA      AA    SSSSSS  UU      UU  DD      DD  FFFFFFFF    RRRRRRRR    LL
BBBBBBBB     AA      AA    SSSSSS  UU      UU  DD      DD  FFFFFFFF    RRRRRRRR    LL
BB      BB   AAAAAAAAAA          SS UU      UU  DD      DD  FF          RR  RR      LL
BB      BB   AAAAAAAAAA          SS UU      UU  DD      DD  FF          RR  RR      LL
BB      BB   AA      AA          SS UU      UU  DD      DD  FF          RR    RR    LL                ....
BB      BB   AA      AA          SS UU      UU  DD      DD  FF          RR    RR    LL                ....
BBBBBBBB     AA      AA  SSSSSSSS   UUUUUUUUUU  DDDDDDDD    FF          RR      RR  LLLLLLLLLL      ....
BBBBBBBB     AA      AA  SSSSSSSS   UUUUUUUUUU  DDDDDDDD    FF          RR      RR  LLLLLLLLLL      ....
```

```
LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE BAS$UDF_RL (                      ! BASIC list-directed input, UDF level
;     2          0002  0                 IDENT = '1-075'          ! File: BASUDFRL.B32 Edit:MDL1075
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                     *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    18          0018  1 !*  TRANSFERRED.                                                              *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    22          0022  1 !*  CORPORATION.                                                              *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1 !
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 ! FACILITY:     BASIC support library - not user callable
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !
;    36          0036  1 !       This module implements BASIC read list-directed I/O statement
;    37          0037  1 !       at the UDF level of abstraction.  This module calls the list-
;    38          0038  1 !       directed record routines at the record level to read a record.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:  User access mode, reentrant AST level or not
;    41          0041  1 !
;    42          0042  1 ! AUTHOR: Donald G. Petersen,   CREATION DATE: 23-MAR-78
;    43          0043  1 !
;    44          0044  1 ! MODIFIED BY:
;    45          0045  1 !
;    46          0046  1 !       DGP, 23-MAR-78 : VERSION 0
;    47          0047  1 ! 1     - original
;    48          0048  1 ! 1-02  - Change to JSB linkages.  DGP 14-Nov-78
;    49          0049  1 ! 1-004 - Update copyright notice and add device names to REQUIRE
;    50          0050  1 !         files.  JBS 29-NOV-78
;    51          0051  1 ! 1-005 - Change REQUIRE file names from FOR... to OTS...  JBS 07-DEC-78
;    52          0052  1 ! 1-006 - Change to new statement types for INPUT LINE and LINPUT. DGP
;    53          0053  1 !         08-Dec-78
;    54          0054  1 ! 1-007 - Change UDF_RL1 to use dispatch tables to get to REC level.  DGP
;    55          0055  1 !         19-Dec-78
;    56          0056  1 ! 1-008 - Add the necessary functionality to get INPUT LINE properly.  DGP
;    57          0057  1 !         19-Dec-78
```

```
;    58          0058  1 ! 1-009 - Bug fix.  DGP 20-Dec-78
;    59          0059  1 ! 1-010 - Add support for longwords.  DGP 28-Dec-78
;    60          0060  1 ! 1-011 - Add error signal to UDF_WL1 (BAS$K_ILLNUM).  DGP 28-Dec-78
;    61          0061  1 ! 1-012 - Fix bug in Input integer (word).  DGP 02-Jan-79
;    62          0062  1 ! 1-013 -  Change ISB$A_BUF_PTR, BUF_BEG, BUF_END to LUB. DGP 05-Jan-79
;    63          0063  1 ! 1-014 - Make some "cleanup" edits based on the code review.
;    64          0064  1 !         JBS for DGP.  09-JAN-1979
;    65          0065  1 ! 1-015 - Correct some typos.  JBS 10-JAN-1979
;    66          0066  1 ! 1-016 - Expand on some comments.  DGP 15-Jan-79
;    67          0067  1 ! 1-017 - Add code to handle ^Z for INPUT LINE properly.  DGP 15-Jan-79
;    68          0068  1 ! 1-018 - Fix bug in returning text string from GETFIELD. DGP 16-Jan-79
;    69          0069  1 ! 1-019 - Change SIGNAL to STOP for ILLNUM in GETFIELD.  DGP 26-Jan-79
;    70          0070  1 ! 1-020 - Use BASIOERR.REQ to define the I/O error codes.  JBS 20-FEB-1979
;    71          0071  1 ! 1-021 - Modify GETFIELD to strip off leading and trailing spaces and tabs
;    72          0072  1 !         from unquoted strings.  DGP 23-Feb-79
;    73          0073  1 ! 1-022 - Change update of BUF_PTR for text in GETFIELD.  DGP 06-Mar-79
;    74          0074  1 ! 1-023 - Strip all leading spaces and tabs from any text string before check-
;    75          0075  1 !         ing for delimiting quotes.  DGP 15-Mar-79
;    76          0076  1 ! 1-024 - Change PRINT_POS to longword.  DGP 19-Mar-79
;    77          0077  1 ! 1-025 - Don't allow semicolon as numeric field separator on Input.  DGP
;    78          0078  1 !         02-Apr-79
;    79          0079  1 ! 1-026 - If this is not a terminal device, then ignore the prompt.  06-Apr-79
;    80          0080  1 !         DGP
;    81          0081  1 ! 1-027 - Change call to BAS$$STOP to BAS$$STOP_IO.  DGP 16-Apr-79
;    82          0082  1 ! 1-028 - Change a few error messages.  DGP 07-May-79
;    83          0083  1 ! 1-029 - Change OTS$S to STR$.  JBS 23-MAY-1979
;    84          0084  1 ! 1-030 - BAS$$UDF_RL1 returns a status.  DGP 06-Jun-79
;    85          0085  1 ! 1-031 - Fix up BAS$$UDF_RL1 to support MAT INPUT.  DGP 14-Jun-79
;    86          0086  1 ! 1-032 - Use language-specific dispatch tables.  JBS 26-JUN-1979
;    87          0087  1 ! 1-033 - Improve the comments.  DGP  28-Jun-79
;    88          0088  1 ! 1-034 - Use ISB symbols for dispatch tables.  JBS 12-JUL-1979
;    89          0089  1 ! 1-035 - Change calls to STR$COPY.  JBS 16-JUL-1979
;    90          0090  1 ! 1-036 - Change from FOR$ input conversion routines to OTS$.  DGP 17-Jul-79
;    91          0091  1 ! 1-037 - Remove reference to BAS$$SIGDIS_ERR.  JBS 01-AUG-1979
;    92          0092  1 ! 1-038 - Set "don't round" flag for single precision floating when calling
;    93          0093  1 !         the input conversion routine.  DGP 07-Aug-79
;    94          0094  1 ! 1-039 - UDF_RL0 should dispatch to the REC level.  DGP 07-Aug-79
;    95          0095  1 ! 1-040 - Set the prompt buffer size to 0 for MAT INPUT if REC level returns
;    96          0096  1 !         a failure.  DGP 07-Aug-79
;    97          0097  1 ! 1-041 - Strip off leading and trailing nulls from input.  DGP 29-Aug-79
;    98          0098  1 ! 1-042 - Unconditionally clear the prompt buffer after every GET.  DGP 03-Sep-79
;    99          0099  1 ! 1-043 - Switch the order of K_CRLF.  DGP 05-Sep-79
;   100          0100  1 ! 1-044 - Increase K_WORK_STR_LEN to 512.  DGP 10-Sep-79
;   101          0101  1 ! 1-045 - Fix bug in INPUT longwords with tabs and spaces.  DGP 10-Sep-79
;   102          0102  1 ! 1-046 - Only look at low byte of RAB$L_STV for terminator.  DGP 18-Sep-79
;   103          0103  1 ! 1-047 - Clear LUB$L_PRINT_POS just before the GET is done.  DGP 18-Sep-79
;   104          0104  1 ! 1-048 - Prompting should be using LUB$B_PRINT_POS from LUB$A_BUDDY_PTR so
;   105          0105  1 !         that CCPOS picks up the right value.  DGP 18-Sep-79
;   106          0106  1 ! 1-049 - Check for comma after quoted string.  DGP 09-Oct-79
;   107          0107  1 ! 1-050 - Include MAT LINPUT with those statement types which want to
;   108          0108  1 !         read an entire line.  DGP 12-Oct-79
;   109          0109  1 ! 1-051 - Another attempt at handling quoted strings properly.  DGP 18-Oct-79
;   110          0110  1 ! 1-052 - Fix bug of input string that is only spaces, tabs, or nulls.
;   111          0111  1 !         DGP 29-Oct-79
;   112          0112  1 ! 1-053 - Pass the scale factor to the conversion routine.  DGP 25-Nov-79
;   113          0113  1 ! 1-054 - Set V_EXP_LETTER for OTS$CVT_T_D.  DGP 04-DEC-79
;   114          0114  1 ! 1-055 - Correct improper register declaration for scaling.  DGP 18-Dec-79
```

```
;  115          0115  1 ! 1-056 - Call MTH$DINT_R3 instead of MTH$DFLOOR_R3 for scaling.  DGP 19-Dec-79
;  116          0116  1 ! 1-057 - Signal DATA FORMAT ERROR instead of ILLEGAL NUMBER.  DGP  21-Jan-80
;  117          0117  1 ! 1-058 - If this is READ or MAT READ then update the data pointer before
;  118          0118  1 !         doing the conversion so that we are pointing at the next data
;  119          0119  1 !         element.  DGP 22-Jan-80
;  120          0120  1 ! 1-059 - Pick up escape sequences from RMS for INPUT LINE.  DGP 21-Feb-80
;  121          0121  1 ! 1-060 - Do not set the cursor position unconditionally to zero.  DGP 04-Mar-80
;  122          0122  1 ! 1-061 - If the terminator is an escape (altmode) and the terminating
;  123          0123  1 !         sequence is of length 1, then transfer the escape character for
;  124          0124  1 !         INPUT LINE.  RMS does not supply it at the end of the data anymore.
;  125          0125  1 !         DGP 31-Mar-80
;  126          0126  1 ! 1-062 - Fix the problem with inputing (READ,INPUT.....)
;  127          0127  1 !         "abc"123,"xyz"  this should give an error because of 123.  FM 25-SEP-80
;  128          0128  1 ! 1-063 - Enable INPUT and kind to take an input longer than K_STR_LEN bytes.
;  129          0129  1 !         Terminal I/O is still restricted to 512 bytes.  FM 25-SEP-80
;  130          0130  1 !         61A and 61B were put in the same packet.
;  131          0131  1 ! 1-064 - Fix problem in above change.  A GTRU should be a GTR.  DGP 03-Feb-1981.
;  132          0132  1 ! 1-065 - Change some occurences of CCB[LUB$L_PRINT_POS] TO TEMP_CCB[LUB$L_PRINT_POS].
;  133          0133  1 !         Also, INPUT should cancel any outstanding PRINT formatcharacter
;  134          0134  1 !         unless the INPUT was terminated by an escape. PLL 12-Jun-81
;  135          0135  1 ! 1-066 - A case statement in GETFIELD modified to always return a value
;  136          0136  1 !         so that the BLISS compiler does not give an error message.
;  137          0137  1 !         PLL 1-Jul-81
;  138          0138  1 ! 1-067 - 64k bytes of data causes a premature "out of data" message because
;  139          0139  1 !         SCANC length is limited to 16 bits.  Make sure the length always looks
;  140          0140  1 !         = or < 64k to GETFIELD.  PLL 23-Jul-81
;  141          0141  1 ! 1-068 - Add support for byte, g floating, and h floating.  PLL 24-Aug-81
;  142          0142  1 ! 1-069 - Add support for packed decimal.  PLL 5-Oct-81
;  143          0143  1 ! 1-070 - More edits for packed decimal.  PLL 29-Dec-81
;  144          0144  1 ! 1-071 - Correct a typo in range check on byte.  PLL 9-Mar-1982
;  145          0145  1 ! 1-072 - Before calling BAS$CVT_T_P, check the decimal rounding/truncation
;  146          0146  1 !         bit in the Basic frame.
;  147          0147  1 ! 1-073 - Add support for ANSI INPUT.  Although input is always from a
;  148          0148  1 !         terminal, errors should cause the entire statement to be re-
;  149          0149  1 !         started not just the specific element.  This means that $GET
;  150          0150  1 !         occurs at the 0 level rather than level 1.  PLL 29-Jul-1982
;  151          0151  1 ! 1-074 - ANSI INPUT of a single element should signal 'too little data',
;  152          0152  1 !         not supply the default for the data type.  PLL 27-Sep-1982
;  153          0153  1 ! 1-075 - allow for terminator space when allocating space for WORK_STR.
;  154          0154  1 !         MDL 25-Apr-1984
;  155          0155  1 !--
```

```
;  157          0156  1
;  158          0157  1 !
;  159          0158  1 ! SWITCHES:
;  160          0159  1 !
;  161          0160  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;  162          0161  1 !
;  163          0162  1 !       LINKAGES:
;  164          0163  1 !
;  165          0164  1
;  166          0165  1 REQUIRE 'RTLIN:OTSLNK';         ! define all linkages
;  167          0594  1
;  168          0595  1 !
;  169          0596  1 ! TABLE OF CONTENTS:
;  170          0597  1 !
;  171          0598  1
;  172          0599  1 FORWARD ROUTINE
;  173          0600  1
;  174          0601  1     !
;  175          0602  1     ! UDF routines
;  176          0603  1     !
;  177          0604  1     BAS$$UDF_RL0: JSB_UDF0 NOVALUE,
;  178          0605  1     BAS$$UDF_RL1: CALL_CCB,
;  179          0606  1     UDF_RL1_HANDLER,
;  180          0607  1     BAS$$UDF_RL9: JSB_UDF9 NOVALUE,
;  181          0608  1
;  182          0609  1     !
;  183          0610  1     ! routine used by  BAS$$UDF_RL1
;  184          0611  1     !
;  185          0612  1     GETFIELD: CALL_CCB;
;  186          0613  1 !
;  187          0614  1 ! INCLUDE FILES:
;  188          0615  1 !
;  189          0616  1 REQUIRE 'RTLML:BASPAR';         ! BASIC intermodule parameters
;  190          0638  1 REQUIRE 'RTLIN:BASFRAME';       ! BASIC frame offsets
;  191          0841  1 REQUIRE 'RTLML:OTSISB';         ! I/O statement block
;  192          1009  1 REQUIRE 'RTLML:OTSLUB';         ! Logical Unit Block
;  193          1149  1 REQUIRE 'RTLIN:OTSMAC';         ! Macros
;  194          1343  1 REQUIRE 'RTLIN:RTLPSECT';       ! Define DECLARE_PSECTS macro
;  195          1438  1 REQUIRE 'RTLIN:BASIOERR';       ! Define I/O error codes.
;  196          1491  1 LIBRARY 'RTLSTARLE';            ! STARLET library for macros and symbols
;  197          1492  1
;  198          1493  1 !
;  199          1494  1 ! MACROS:
;  200          1495  1 !
;  201          1496  1 !       NONE
;  202          1497  1 !
;  203          1498  1 !
;  204          1499  1 ! EQUATED SYMBOLS:
;  205          1500  1 !
;  206          1501  1
;  207          1502  1 LITERAL
;  208          1503  1
;  209          1504  1     K_WORK_STR_LEN = 512,           ! length of work area for parsing input.
;  210          1505  1     K_NULL         = 0,             ! types of constants which may appear in input record
;  211          1506  1     K_CR           = %X'0D',        ! ASCII <cr>
;  212          1507  1     K_ESC          = %X'1B',        ! ASCII <esc>
;  213          1508  1     K_SP           = %X'20',        ! ASCII <sp>
```

```
;  214          1509  1     K_TAB       = 9;                          ! ASCII TAB
;  215          1510  1
;  216          1511  1 BUILTIN
;  217          1512  1     CVTSP,
;  218          1513  1     SCANC;
;  219          1514  1
;  220          1515  1 !
;  221          1516  1 ! PSECT declarations
;  222          1517  1 !
;  223          1518  1     DECLARE_PSECTS (BAS);                      ! declare PSECTS for BAS$ facility
;  224          1519  1
;  225          1520  1 !
;  226          1521  1 ! OWN STORAGE:
;  227          1522  1 !       NONE
;  228          1523  1
;  229          1524  1 !
;  230          1525  1 ! EXTERNAL REFERENCES:
;  231          1526  1 !
;  232          1527  1
;  233          1528  1 EXTERNAL LITERAL
;  234          1529  1     BAS$K_DATFORERR:UNSIGNED (8),              ! Data format error
;  235          1530  1     BAS$K_ILLNUM:UNSIGNED (8),                 ! Illegal number
;  236          1531  1     BAS$K_ENDFILDEV:UNSIGNED (8),              ! End of file on device
;  237          1532  1     BAS$K_MAXMEMEXC:UNSIGNED (8),              ! Maximum memory exceeded
;  238          1533  1     BAS$K_PROLOSSOR:UNSIGNED (8),              ! Program lost sorry
;  239          1534  1     BAS$K_TOOLITDAT:UNSIGNED (8);              ! Too little data (ANSI only)
;  240          1535  1
;  241          1536  1 EXTERNAL
;  242          1537  1     BAS$$AA_REC_PR0 : VECTOR,                  ! Dispatch table for REC init.
;  243          1538  1     BAS$$AA_REC_PR1 : VECTOR,                  ! Dispatch table for REC level
;  244          1539  1     OTS$$A_CUR_LUB: ADDRESSING_MODE (GENERAL), ! address of currently active LUB/ISB/RAB
;  245          1540  1     BAS$HANDLER;                               ! just need the address of this
;  246          1541  1
;  247          1542  1 EXTERNAL ROUTINE
;  248          1543  1     MTH$DINT,                                  ! Remove fraction after scaling
;  249          1544  1     BAS$$STOP_IO,                              ! signal fatal errors
;  250          1545  1     BAS$$SIGNAL_IO,                            ! signal an error
;  251          1546  1     LIB$CVTDF,                                 ! convert double to floating
;  252          1547  1     STR$COPY_DX,                               ! Copy a string by descriptor
;  253          1548  1     BAS$OUT_T_DX_S:  NOVALUE,                  ! output a text string
;  254          1549  1     BAS$CVT_T_P,                               ! convert text to packed decimal
;  255          1550  1     !
;  256          1551  1     ! conversion routines
;  257          1552  1     !
;  258          1553  1     OTS$CVT_TI_L,                              ! convert ASCII to internal 32 bit integer
;  259          1554  1     OTS$CVT_T_D,                               ! convert ASCII to internal double precision
;  260          1555  1     OTS$CVT_T_G,                               ! convert ASCII to internal g floating
;  261          1556  1     OTS$CVT_T_H,                               ! convert ASCII to internal h floating
;  262          1557  1     !
;  263          1558  1     ! record level routines for list-directed input
;  264          1559  1     !
;  265          1560  1     BAS$$REC_RSL0: JSB_REC0 NOVALUE,           ! initialize Input record level
;  266          1561  1     BAS$$REC_RSL9: JSB_REC9 NOVALUE,           ! end of Input record level
;  267          1562  1     LIB$GET_VM,                                ! get virtual memory
;  268          1563  1     LIB$FREE_VM,                               ! free virtual memory
;  269          1564  1     LIB$MATCH_COND;                            ! match the condition value
;  270          1565  1
```

```
;  272          1566  1 GLOBAL ROUTINE BAS$$UDF_RL0 (
;  273          1567  1         FORMAT_ADR
;  274          1568  1         ): JSB_UDF0 NOVALUE =
;  275          1569  1
;  276          1570  1 !++
;  277          1571  1 ! FUNCTIONAL DESCRIPTION:
;  278          1572  1 !
;  279          1573  1 !       Perform UDF level read list-directed I/O initialization.
;  280          1574  1 !       Initialize module "own" storage in the ISB.
;  281          1575  1 !       Call record level processor to get first input record.
;  282          1576  1 !
;  283          1577  1 ! FORMAL PARAMETERS:
;  284          1578  1 !
;  285          1579  1 !       FORMAT_ADR.rl.r         Not used
;  286          1580  1 !
;  287          1581  1 ! IMPLICIT INPUTS:
;  288          1582  1 !
;  289          1583  1 !       OTS$$A_CUR_LUB          Pointer to current logical unit block (LUB)
;  290          1584  1 !
;  291          1585  1 ! IMPLICIT OUTPUTS:
;  292          1586  1 !
;  293          1587  1 !       NONE
;  294          1588  1 !
;  295          1589  1 ! ROUTINE VALUE:
;  296          1590  1 ! COMPLETION CODES:
;  297          1591  1 !
;  298          1592  1 !       NONE
;  299          1593  1 !
;  300          1594  1 ! SIDE EFFECTS:
;  301          1595  1 !
;  302          1596  1 !       NONE
;  303          1597  1 !
;  304          1598  1 !--
;  305          1599  1
;  306          1600  2     BEGIN
;  307          1601  2     EXTERNAL REGISTER
;  308          1602  2         CCB: REF BLOCK[, BYTE];
;  309          1603  2
;  310          1604  2     !+
;  311          1605  2     ! Call record level routine to read the first record.
;  312          1606  2     ! The buffer pointers are initialized based on whether the device is
;  313          1607  2     ! a terminal or not
;  314          1608  2     !-
;  315          1609  2
;  316          1610  2     !+
;  317          1611  2     ! If this is an ANSI INPUT, the RECO level will ask for input.  So
;  318          1612  2     ! put out the standard prompt.  Note:  ANSI has no files, so INPUT
;  319          1613  2     ! will always be from a terminal.
;  320          1614  2     !-
;  321          1615  2
;  322          1616  2         IF .CCB [LUB$V_ANSI]
;  323          1617  2         THEN
;  324          1618  3             BEGIN
;  325          1619  3             LOCAL
;  326          1620  3                 TDSC: VECTOR [2];
;  327          1621  3             BIND
;  328          1622  3             D_PROMPT = UPLIT ('? ');
```

```
;  329          1623  3              TDSC[0] = %CHARCOUNT ('? ');
;  330          1624  3              TDSC[1] = D_PROMPT;
;  331          1625  3              BAS$OUT_T_DX_S(TDSC);
;  332          1626  2              END;
;  333          1627  2
;  334          1628  2      JSB_REC0 (BAS$$AA_REC_PR0 + .BAS$$AA_REC_PR0 [.CCB [ISB$B_STTM_TYPE] - ISB$K_BASSTTYLO + 1]);
;  335          1629  2
;  336          1630  1      END;


                                                        .TITLE  BAS$$UDF_RL
                                                        .IDENT  \1-075\

                                                        .PSECT  _BAS$CODE,NOWRT,  SHR,  PIC,2

                                    00  00  20  3F  00000 P.AAA:  .ASCII  \? \<0><0>                                    ;

                                                    D_PROMPT=           P.AAA
                                                        .EXTRN  BAS$K_DATFORERR
                                                        .EXTRN  BAS$K_ILLNUM, BAS$K_ENDFILDEV
                                                        .EXTRN  BAS$K_MAXMEMEXC
                                                        .EXTRN  BAS$K_PROLOSSOR
                                                        .EXTRN  BAS$K_TOOLITDAT
                                                        .EXTRN  BAS$$AA_REC_PR0
                                                        .EXTRN  BAS$$AA_REC_PR1
                                                        .EXTRN  OTS$$A_CUR_LUB, BAS$HANDLER
                                                        .EXTRN  MTH$DINT, BAS$$STOP_IO
                                                        .EXTRN  BAS$$SIGNAL_IO, LIB$CVTDF
                                                        .EXTRN  STR$COPY_DX, BAS$OUT_T_DX_S
                                                        .EXTRN  BAS$CVT_T_P, OTS$CVT_TI_L
                                                        .EXTRN  OTS$CVT_T_D, OTS$CVT_T_G
                                                        .EXTRN  OTS$CVT_T_H, BAS$$REC_RSL0
                                                        .EXTRN  BAS$$REC_RSL9, LIB$GET_VM
                                                        .EXTRN  LIB$FREE_VM, LIB$MATCH_COND

                                        5E              08  C2 00000 BAS$$UDF_RL0::
                                                                      SUBL2   #8, SP                                    ; 1566
                                    11      A1  AB              04  E1 00003          BBC     #4, -95(CCB), 1$          ; 1616
                                            6E              02  D0 00008          MOVL    #2, TDSC                  ; 1623
                                                04  AE      EE  AF 9E 0000B          MOVAB   D_PROMPT, TDSC+4          ; 1624
                                                    5E          DD 00010          PUSHL   SP                        ; 1625
                                        00000000G   00          01  FB 00012          CALLS   #1, BAS$OUT_T_DX_S        ;
                                                    50      FF71    CB 9A 00019 1$:      MOVZBL  -143(CCB), R0             ; 1628
                                                    50  00000000G0040   D0 0001E          MOVL    BAS$$AA_REC_PR0-104[R0], R0
                                                        00000000G0040   16 00026          JSB     BAS$$AA_REC_PR0[R0]
                                                    5E              08  C0 0002D          ADDL2   #8, SP                    ; 1630
                                                                    05  00030          RSB                               ;

; Routine Size:  49 bytes,    Routine Base:  _BAS$CODE + 0004
```

```
;  338          1631  1 GLOBAL ROUTINE BAS$$UDF_RL1 (
;  339          1632  1         ELEM_TYPE,
;  340          1633  1         ELEM_SIZE,
;  341          1634  1         ELEM_ADR,
;  342          1635  1         FORMAT
;  343          1636  1         )
;  344          1637  1         : CALL_CCB =
;  345          1638  1
;  346          1639  1 !++
;  347          1640  1 ! FUNCTIONAL DESCRIPTION:
;  348          1641  1 !
;  349          1642  1 !       Return the next input value to the user I/O list element.
;  350          1643  1 !       The value obtained from the input record is converted to
;  351          1644  1 !       the type of the list element.
;  352          1645  1 !
;  353          1646  1 ! FORMAL PARAMETERS:
;  354          1647  1 !
;  355          1648  1 !       ELEM_TYPE.rlu.v         Type code of user I/O list element
;  356          1649  1 !       ELEM_SIZE.rlu.v         Size of the list element
;  357          1650  1 !       ELEM_ADR.rlu.r          Adr of where to store the element
;  358          1651  1 !                               Points to a descriptor for a string
;  359          1652  1 !       FORMAT.rlu.v            Format character following a Prompt string
;  360          1653  1 !
;  361          1654  1 ! IMPLICIT INPUTS:
;  362          1655  1 !
;  363          1656  1 !       OTS$$A_CUR_LUB          Pointer to current logical unit block (LUB)
;  364          1657  1 !       LUB$L_PRINT_POS         Internal cursor position
;  365          1658  1 !       LUB$V_UNIT_0            flag to indicate terminal on unit 0
;  366          1659  1 !
;  367          1660  1 ! IMPLICIT OUTPUTS:
;  368          1661  1 !
;  369          1662  1 !       LUB$L_PRINT_POS         internal cursor position
;  370          1663  1 !       RAB$B_PSZ               size of the Prompt buffer
;  371          1664  1 !
;  372          1665  1 ! ROUTINE VALUE:
;  373          1666  1 ! COMPLETION CODES:
;  374          1667  1 !
;  375          1668  1 !       NONE
;  376          1669  1 !
;  377          1670  1 ! SIDE EFFECTS:
;  378          1671  1 !
;  379          1672  1 !       SIGNALS various errors for input incompatibility and not enough
;  380          1673  1 !       input data.
;  381          1674  1 !       If this is not a terminal device, then ignore any prompts.
;  382          1675  1 !
;  383          1676  1 !       NOTICE : All terminal device files are allocated the static buffer for
;  384          1677  1 !                parsing, i.e. no VM is allocated for them (because at the
;  385          1678  1 !                time this routine is called we don't know how large of input
;  386          1679  1 !                we have!!). This means that the maximum terminal device input
;  387          1680  1 !                is K_WORK_STR_LEN, anything over this will write over the
;  388          1681  1 !                stack.
;  389          1682  1 !--
;  390          1683  1
;  391          1684  1     !+
;  392          1685  1     ! Be aware that there are two exit points in this routine.  One is from
;  393          1686  1     ! the Prompt handling section and the other is from the Input handling section
;  394          1687  1     !-
```

```
;  395          1688  1
;  396          1689  2      BEGIN
;  397          1690  2      MAP
;  398          1691  2          ELEM_ADR: REF VECTOR;
;  399          1692  2      LOCAL
;  400          1693  2          BYTES_NEEDED: INITIAL(0),          ! workspace needed
;  401          1694  2          WORKSPACE: VECTOR [ K_WORK_STR_LEN , BYTE ],! if input is K_WORK_STR_LEN or less
;  402          1695  2                                             ! use this space through CHARCONS.
;  403          1696  2          CHARCONS: REF VECTOR [ , BYTE ],   ! The space where the parsing of input takes place.
;  404          1697  2          D_VALUE: VECTOR[4],                ! holds binary equivalent of input char.
;  405          1698  2                                             ! for numerics
;  406          1699  2          TEMP_CCB : REF BLOCK [, BYTE],     ! temporary CCB
;  407          1700  2          DSC: BLOCK [8,BYTE],               ! Descriptor of parsed element for strings
;  408          1701  2                                             ! need a local descriptor because if a
;  409          1702  2                                             ! static desc. was passed, the values in
;  410          1703  2                                             ! it are not to be changed.
;  411          1704  2          UNWIND_VM_SIZE : VOLATILE,         ! Size of VM allocated.
;  412          1705  2          UNWIND_VM_ADDR : VOLATILE,         ! Address of VM allocated for input buffer
;  413          1706  2                                             ! This buffer is allocated if input size
;  414          1707  2                                             ! is greater than 512.
;  415          1708  2          UNWIND_CCB     : VOLATILE;         ! CCB for the handler.
;  416          1709  2
;  417          1710  2      LITERAL
;  418          1711  2          K_ESC = %X'1B';          ! ASCII for escape
;  419          1712  2
;  420          1713  2      EXTERNAL REGISTER
;  421          1714  2          CCB: REF BLOCK[,BYTE];
;  422          1715  2
;  423          1716  2      BUILTIN
;  424          1717  2          FP;
;  425          1718  2
;  426          1719  2  !+
;  427          1720  2  ! Set up a handler for this routine so in case of unwind we can deallocate VM,
;  428          1721  2  ! if any was allocated.
;  429          1722  2  !-
;  430          1723  2      ENABLE UDF_RL1_HANDLER ( UNWIND_VM_SIZE , UNWIND_VM_ADDR , UNWIND_CCB );
;  431          1724  2
;  432          1725  2  !+
;  433          1726  2  ! determine how much workspace is needed.  this is the number of bytes in
;  434          1727  2  ! the buffer plus the number of bytes in the terminator.
;  435          1728  2  !-
;  436          1729  3      BYTES_NEEDED = ( (.CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR]) +
;  437          1730  4                      (SELECTONEU .CCB [RAB$W_STV0] OF
;  438          1731  4                       SET
;  439          1732  4                         [K_ESC]:         .CCB [RAB$W_STV2];
;  440          1733  4                         [K_CR]:          2;
;  441          1734  4                         [OTHERWISE]:     0;
;  442          1735  2                       TES) );
;  443          1736  2  !+
;  444          1737  2  ! If space needed for parsing is greater than K_WORK_STR_LEN then we use VM, otherwise
;  445          1738  2  ! we use the static storage allocated in WORKSPACE.
;  446          1739  2  !-
;  447          1740  2      IF .BYTES_NEEDED GTR K_WORK_STR_LEN
;  448          1741  2      THEN
;  449          1742  3          BEGIN
;  450          1743  3          UNWIND_VM_SIZE = .BYTES_NEEDED;
;  451          1744  3          UNWIND_CCB = .CCB;
```

```
;  452          1745  3          IF NOT LIB$GET_VM ( UNWIND_VM_SIZE , UNWIND_VM_ADDR ) THEN BAS$$STOP_IO (BAS$K_MAXMEMEXC);
;  453          1746  3          CHARCONS = .UNWIND_VM_ADDR;
;  454          1747  3          END
;  455          1748  2      ELSE
;  456          1749  2          CHARCONS = WORKSPACE;
;  457          1750  2      !+
;  458          1751  2      ! Load up TEMP_CCB with a pointer to the complementary data base for PRINT.
;  459          1752  2      !-
;  460          1753  2      TEMP_CCB = .CCB [LUB$A_BUDDY_PTR];
;  461          1754  2
;  462          1755  2      IF .FORMAT GTR 0
;  463          1756  2      THEN
;  464          1757  2          BEGIN
;  465          1758  2
;  466          1759  2          !+
;  467          1760  2          ! Check to see if this is a terminal device.  If it is, then process the
;  468          1761  2          ! prompt; otherwise, just return.
;  469          1762  2          !-
;  470          1763  2
;  471          1764  2          IF .CCB [LUB$V_TERM_DEV]
;  472          1765  2          THEN
;  473          1766  3              BEGIN
;  474          1767  3
;  475          1768  3          !+
;  476          1769  3          ! Prompt
;  477          1770  3          !-
;  478          1771  3
;  479          1772  3          LOCAL
;  480          1773  3              RDSC: BLOCK [8, BYTE];      ! Resultant descriptor from Prompt processing
;  481          1774  3          LITERAL
;  482          1775  3              K_PRINT_ZONE_SZ = 14,       ! Print zone size
;  483          1776  3              K_CRLF = %X'0A0D';          ! ASCII codes for carriage return-line feed
;  484          1777  3
;  485          1778  3          RDSC[DSC$A_POINTER] = .CCB[RAB$L_PBF] + .CCB[RAB$B_PSZ];
;  486          1779  3
;  487          1780  3          !+
;  488          1781  3          ! adjust the internal cursor position and the resultant string
;  489          1782  3          ! length as determined by the data type and the format character
;  490          1783  3          !-
;  491          1784  3
;  492          1785  3          CASE .FORMAT
;  493          1786  3          FROM BAS$K_SEMI_FORM TO BAS$K_NO_FORM OF
;  494          1787  3          SET
;  495          1788  3          [BAS$K_SEMI_FORM]:
;  496          1789  4              BEGIN
;  497          1790  4              CCB[ISB$V_P_FORM_CH] = BAS$K_SEMI_FORM;
;  498          1791  4              RDSC[DSC$W_LENGTH] = .ELEM_SIZE;
;  499          1792  4              TEMP_CCB [LUB$L_PRINT_POS] = .ELEM_SIZE + .TEMP_CCB [LUB$L_PRINT_POS];
;  500          1793  3              END;
;  501          1794  3          [BAS$K_COMMA_FOR]:
;  502          1795  4              BEGIN
;  503          1796  4              CCB[ISB$V_P_FORM_CH] = BAS$K_COMMA_FOR;
;  504          1797  7              RDSC[DSC$W_LENGTH] = .ELEM_SIZE + (K_PRINT_ZONE_SZ - ((.TEMP_CCB [LUB$L_PRINT_POS] + .ELEM_SIZE)
;  505          1798  4                  MOD K_PRINT_ZONE_SZ));
;  506          1799  4              TEMP_CCB [LUB$L_PRINT_POS] = .TEMP_CCB[LUB$L_PRINT_POS] + .RDSC[DSC$W_LENGTH];
;  507          1800  3              END;
;  508          1801  3          [BAS$K_NO_FORM]:
```

```
;  509          1802  5              BEGIN
;  510          1803  5
;  511          1804  5              !+
;  512          1805  5              ! Need to leave room for carriage control
;  513          1806  5              !-
;  514          1807  5
;  515          1808  5              RDSC[DSC$W_LENGTH] = .ELEM_SIZE + 2;
;  516          1809  5              CCB[ISB$V_P_FORM_CH] = BAS$K_NO_FORM;
;  517          1810  5              TEMP_CCB[CUB$L_PRINT_POS] = 0;
;  518          1811  4              END;
;  519          1812  4          TES;
;  520          1813  4
;  521          1814  4          !+
;  522          1815  4          ! Set the address for the destination of the Prompt.  Update the RAB
;  523          1816  4          ! Prompt Buffer Size
;  524          1817  4          !-
;  525          1818  4
;  526          1819  4          CCB[RAB$B_PSZ] = .CCB[RAB$B_PSZ] + .RDSC[DSC$W_LENGTH];
;  527          1820  4          RDSC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  528          1821  4          CH$COPY (.ELEM_SIZE, .(.ELEM_ADR+4), ' ', .RDSC[DSC$W_LENGTH], .RDSC[DSC$A_POINTER]);
;  529          1822  4          IF .FORMAT EQL0 BAS$K_NO_FORM
;  530          1823  4          THEN
;  531          1824  4              (.RDSC[DSC$A_POINTER] + .ELEM_SIZE)<0, 16> = K_CRLF;
;  532          1825  3          END;
;  533          1826  3          RETURN 1;
;  534          1827  2          END;
;  535          1828  2
;  536          1829  2      !+
;  537          1830  2      ! This section is concerned with inputting a value
;  538          1831  2      ! GETFIELD will attempt to parse another field out of the INPUT stream based
;  539          1832  2      ! on the data type.  If a data field cannot be found (empty buffer)
;  540          1833  2      ! then a failure
;  541          1834  2      ! status is returned.  If a data field is found then a
;  542          1835  2      !conversion, for numerics,
;  543          1836  2      ! is done and if a conversion error occurs, the error number is put into the
;  544          1837  2      ! LUB.  For a string, the descriptor passed to GETFIELD is updated to point to
;  545          1838  2      ! the parsed string and the length field is updated.
;  546          1839  2      !-
;  547          1840  2
;  548          1841  3      IF NOT (GETFIELD(
;  549          1842  3
;  550          1843  3                  !+
;  551          1844  3                  ! Pass the a reference to a quadword for a numeric quantity and
;  552          1845  3                  ! a pointer to a descriptor for a string
;  553          1846  3                  !-
;  554          1847  3
;  555          1848  4                  (CASE .ELEM_TYPE
;  556          1849  4                  FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
;  557          1850  4                  SET
;  558          1851  4                  [DSC$K_DTYPE_B, DSC$K_DTYPE_W, DSC$K_DTYPE_L, DSC$K_DTYPE_F,
;  559          1852  4                   DSC$K_DTYPE_D, DSC$K_DTYPE_G, DSC$K_DTYPE_H]:
;  560          1853  4                      D_VALUE;
;  561          1854  4                  [DSC$K_DTYPE_T, DSC$K_DTYPE_P] :
;  562          1855  4                      DSC;
;  563          1856  4                  [INRANGE, OUTRANGE]:
;  564          1857  4                      !+
;  565          1858  4                      ! Data types which are not yet supported
```

```
;  566          1859  4                    !-
;  567          1860  4                    0
;  568          1861  4                TES
;  569          1862  3                ),
;  570          1863  3            .ELEM_TYPE, .CHARCONS))
;  571          1864  2        THEN
;  572          1865  3            BEGIN
;  573          1866  3            !+
;  574          1867  3            ! Try to get another record.  Device type checking (forcible or nonforcible) is performed at
;  575          1868  3            ! the REC level before a GET is attempted.
;  576          1869  3            !-
;  577          1870  3            IF .CCB[LUB$V_UNIT_0] AND NOT .CCB [LUB$V_ANSI]
;  578          1871  3            THEN
;  579          1872  3                !+
;  580          1873  3                ! Insert the BASIC default prompt if on unit 0
;  581          1874  3                !-
;  582          1875  3
;  583          1876  4                BEGIN
;  584          1877  4                LOCAL
;  585          1878  4                    TDSC: VECTOR [2];
;  586          1879  4                BIND
;  587          1880  4                    D_PROMPT = UPLIT ('? ');
;  588          1881  4                TDSC[0] = %CHARCOUNT ('? ');
;  589          1882  4                TDSC[1] = D_PROMPT;
;  590          1883  4                BAS$OUT_T_DX_S(TDSC);
;  591          1884  3                END;
;  592          1885  3
;  593          1886  3            !+
;  594          1887  3            ! Dispatch to the appropriate REC level routine.  If INPUT then get
;  595          1888  3            ! another record.  If READ then signal an error - should not be out
;  596          1889  3            ! of data. If this is a MAT INPUT, try to get another record and pass the status
;  597          1890  3            ! back to the UPI level.  Status is determined by whether the current
;  598          1891  3            ! record ends with a continuation character.  Clear LUB$L_PRINT_POS thru
;  599          1892  3            ! BUDDY_PTR so that this INPUT will not affect later PRINTs or prompting
;  600          1893  3            ! if there is an error on this GET.
;  601          1894  3            ! NOTE:  There is a RETURN here in the middle of the routine.
;  602          1895  3            !-
;  603          1896  3
;  604          1897  4            IF (NOT (JSB_REC1 (BAS$$AA_REC_PR1 + .BAS$$AA_REC_PR1[.CCB[ISB$B_STTM_TYPE] - ISB$K_BASSTTYLO + 1]))
;  605          1898  3            THEN
;  606          1899  3        !+
;  607          1900  3        ! Clear the Prompt buffer which has been loaded in case another GET was going to
;  608          1901  3        ! be done.  If it is not cleared, then I/O END will print it out (10 INPUT 'foo'
;  609          1902  3        ! ).  MAT INPUT is different, because the RTL asks for more data if it is avail-
;  610          1903  3        ! able.  The other types of Input demand more data.  Therefore, the GET for MAT
;  611          1904  3        ! INPUT is only done if the continuation flag is set signifying that the last
;  612          1905  3        ! record ended in an '&'.
;  613          1906  3        !-
;  614          1907  4                BEGIN
;  615          1908  4                CCB [RAB$B_PSZ] = 0;
;  616          1909  4                RETURN 0;
;  617          1910  3                END;
;  618          1911  3        !+
;  619          1912  3        ! Unconditionally clear the prompt buffer so that a RESUME with no line number
;  620          1913  3        ! which restarts an INPUT statement will not keep concatenating prompt strings.
;  621          1914  3        !-
;  622          1915  3            CCB [RAB$B_PSZ] = 0;
```

```
;  623          1916  3          !+
;  624          1917  3          ! Now that another record has been gotten, call GETFIELD again and ignore
;  625          1918  3          ! the return status because it is assumed that failure to return something
;  626          1919  3          ! is impossible.
;  627          1920  3          !-
;  628          1921  3
;  629          1922  3          GETFIELD(
;  630          1923  3                  (CASE .ELEM_TYPE
;  631          1924  4                  FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
;  632          1925  4                  SET
;  633          1926  4                  [DSC$K_DTYPE_B, DSC$K_DTYPE_W, DSC$K_DTYPE_L, DSC$K_DTYPE_F,
;  634          1927  4                   DSC$K_DTYPE_D, DSC$K_DTYPE_G, DSC$K_DTYPE_H]:
;  635          1928  4                      D_VALUE;
;  636          1929  4                  [DSC$K_DTYPE_T, DSC$K_DTYPE_P]:
;  637          1930  4                      DSC;
;  638          1931  4                  [INRANGE, OUTRANGE]:
;  639          1932  4                      !+
;  640          1933  4                      ! Data types which are not yet supported
;  641          1934  4                      !-
;  642          1935  4                      0
;  643          1936  4                  TES
;  644          1937  4                  ),
;  645          1938  3              .ELEM_TYPE, .CHARCONS)
;  646          1939  3          END;
;  647          1940  2
;  648          1941  2
;  649          1942  2          !+
;  650          1943  2          ! Store the converted Input data into its new home based on the data type
;  651          1944  2          !-
;  652          1945  2
;  653          1946  2          CASE .ELEM_TYPE
;  654          1947  2          FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
;  655          1948  2          SET
;  656          1949  2          [INRANGE, OUTRANGE]:
;  657          1950  2          !+
;  658          1951  2          ! Data types which are not supported
;  659          1952  2          !-
;  660          1953  2          0;
;  661          1954  2          [DSC$K_DTYPE_B]:
;  662          1955  2              !+
;  663          1956  2              ! Byte
;  664          1957  2              !-
;  665          1958  3              BEGIN
;  666          1959  3              MAP
;  667          1960  3                  ELEM_ADR: REF VECTOR[, BYTE];
;  668          1961  3              ELEM_ADR[0] = .D_VALUE;
;  669          1962  3              END;
;  670          1963  2          [DSC$K_DTYPE_W]:
;  671          1964  2              !+
;  672          1965  2              ! Integer
;  673          1966  2              !-
;  674          1967  3              BEGIN
;  675          1968  3              MAP
;  676          1969  3                  ELEM_ADR: REF VECTOR[, WORD];
;  677          1970  3              ELEM_ADR[0] = .D_VALUE;
;  678          1971  3              END;
;  679          1972  2
```

```
;  680          1973  2      [DSC$K_DTYPE_L, DSC$K_DTYPE_F]:
;  681          1974  2          !+
;  682          1975  2          ! Longword integer or single precision floating point
;  683          1976  2          !-
;  684          1977  2          ELEM_ADR[0] = .D_VALUE;
;  685          1978  2      [DSC$K_DTYPE_D, DSC$K_DTYPE_G]:
;  686          1979  2          !+
;  687          1980  2          ! Double precision floating point or g floating
;  688          1981  2          !-
;  689          1982  2          BEGIN
;  690          1983  2          ELEM_ADR[0] = .D_VALUE[0];
;  691          1984  2          ELEM_ADR[1] = .D_VALUE[1];
;  692          1985  2          END;
;  693          1986  2      [DSC$K_DTYPE_H]:
;  694          1987  2          !+
;  695          1988  2          ! H floating
;  696          1989  2          !-
;  697          1990  2          BEGIN
;  698          1991  2          ELEM_ADR[0] = .D_VALUE[0];
;  699          1992  2          ELEM_ADR[1] = .D_VALUE[1];
;  700          1993  2          ELEM_ADR[2] = .D_VALUE[2];
;  701          1994  2          ELEM_ADR[3] = .D_VALUE[3];
;  702          1995  2          END;
;  703          1996  2      [DSC$K_DTYPE_T]:
;  704          1997  2          !+
;  705          1998  2          ! Character string - ELEM_ADR contains the address of the descriptor
;  706          1999  2          !-
;  707          2000  2          BEGIN
;  708          2001  3          DSC[DSC$A_POINTER] = .CHARCONS;
;  709          2002  3          DSC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  710          2003  3          DSC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  711          2004  3          ! ***** Change to LIB$SCOPY to inhibit signalling ******
;  712          2005  3          STR$COPY_DX (.ELEM_ADR, DSC);
;  713          2006  3          IF .(.DSC[DSC$A_POINTER])<0,8> EQLU BAS$K_CONTROL_Z
;  714          2007  3          THEN
;  715          2008  4              BEGIN
;  716          2009  4
;  717          2010  4              !+
;  718          2011  4              ! This ^Z has been deferred until now so that it could get stored into
;  719          2012  4              ! the users buffer for error handling as required by Basic.  Now is
;  720          2013  4              ! the proper time to signal the error.
;  721          2014  4              !-
;  722          2015  4
;  723          2016  4              CCB[RAB$B_PSZ] = 0;
;  724          2017  4              BAS$$STOP_IO(BAS$K_ENDFILDEV);
;  725          2018  3              END;
;  726          2019  2          END;
;  727          2020  2      [DSC$K_DTYPE_P]:
;  728          2021  2          !+
;  729          2022  2          ! Packed decimal string - ELEM_ADR contains the address of the descriptor
;  730          2023  2          !-
;  731          2024  3          BEGIN
;  732          2025  3          LOCAL
;  733          2026  3              STATUS,
;  734          2027  3              FLAGS,
;  735          2028  3              FMP : REF BLOCK [0,BYTE] FIELD (BSF$FCD);
;  736          2029  3
```

```
;  737          2030  1          LITERAL
;  738          2031  1              V_DONT_ROUND = 1^3;
;  739          2032  1
;  740          2033  1          DSC[DSC$A_POINTER] = .CHARCONS;
;  741          2034  1          DSC[DSC$B_CLASS] = DSC$K_CLASS_S;
;  742          2035  1          DSC[DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  743          2036  1          !+
;  744          2037  1          ! Call a conversion routine which will handle the semantics of converting
;  745          2038  1          ! text to packed decimal.  Pass the decimal round/truncate flag from the
;  746          2039  1          ! Basic frame as the flags parameter.
;  747          2040  1          !-
;  748          2041  1          FMP = .FP;
;  749          2042  1
;  750          2043  1          DO
;  751          2044  1              BEGIN                           ! search for a Basic frame
;  752          2045  1              FMP = .FMP [BSF$A_SAVED_FP];
;  753          2046  1              END
;  754          2047  1          UNTIL (.FMP [BSF$A_HANDLER] EQLA BAS$HANDLER OR
;  755          2048  1                  .FMP EQL 0);
;  756          2049  1
;  757          2050  1          IF (.FMP NEQ 0) AND (.FMP [BSF$W_FCD_FLAGS] AND BSF$M_FCD_RND) NEQ 0
;  758          2051  1          THEN
;  759          2052  1              FLAGS = 0
;  760          2053  1          ELSE
;  761          2054  1              FLAGS = V_DONT_ROUND;       ! set flags according to frame bit
;  762          2055  1
;  763          2056  1          STATUS = BAS$CVT_T_P (DSC, (.ELEM_ADR), .FLAGS);
;  764          2057  1          IF NOT .STATUS THEN BAS$$STOP_IO (BAS$K_DATFORERR);
;  765          2058  1          END
;  766          2059  1      TES;
;  767          2060  1      CCB[RAB$B_PSZ] = 0;
;  768          2061  1      IF (.CCB[RAB$W_STV0] NEQ K_ESC) THEN TEMP_CCB[LUB$V_FORM_CHAR] = 0;
;  769          2062  1  !+
;  770          2063  1  ! If we have allocated VM for the parsing space then deallocate it here.
;  771          2064  1  !-
;  772          2065  1      IF ( .CHARCONS NEQA WORKSPACE )
;  773          2066  1      THEN
;  774          2067  1          BEGIN
;  775          2068  1          IF NOT LIB$FREE_VM ( UNWIND_VM_SIZE , UNWIND_VM_ADDR )
;  776          2069  1          THEN
;  777          2070  1              BEGIN
;  778          2071  1              UNWIND_VM_SIZE = 0;
;  779          2072  1              BAS$$STOP_IO (BAS$K_PROLOSSOR);
;  780          2073  1              END;
;  781          2074  1          END;
;  782          2075  1      RETURN 1;
;  783          2076  1      END;


                                 00035                  .BLKB   3
                          00 00 20 3F 00038 P.AAB:      .ASCII  \? \<0><0>                                  ;
                                                D_PROMPT=           P.AAB

                                     07FC 00000         .ENTRY  BAS$UDF_RL1, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 1631
```

```
                                                                       R10                                   ;
                             5A 0000000G   00    9E 00002          MOVAB   BAS$$STOP_IO, R10                     ;
                             5E     FDD4   CE    9E 00009          MOVAB   -556(SP), SP                          ;
                                           51    D4 0000E          CLRL    BYTES_NEEDED                          ; 1689
                                     08    AE    7C 00010          CLRQ    UNWIND_CCB                            ;
                                     10    AE    D4 00013          CLRL    UNWIND_VM_SIZE                        ;
                             6D     02CD   CF    DE 00016          MOVAL   49$, (FP)                             ;
          52          B4     AB       B0   AB    C3 0001B          SUBL3   -80(CCB), -76(CCB), R2                ; 1729
                             50       0C   AB    3C 00021          MOVZWL  12(CCB), R0                           ; 1730
                             1B            50    B1 00025          CMPW    R0, #27                               ; 1732
                             06            12    00028             BNEQ    1$                                    ;
                             50       0E   AB    3C 0002A          MOVZWL  14(CCB), R0                           ;
                             0C            11    0002E             BRB     3$                                    ;
                             0D            50    B1 00030 1$:      CMPW    R0, #13                               ; 1733
                             05            12    00033             BNEQ    2$                                    ;
                             50            02    D0 00035          MOVL    #2, R0                                ;
                             02            11    00038             BRB     3$                                    ;
                                           50    D4 0003A 2$:      CLRL    R0                                    ; 1734
          51                 52            50    C1 0003C 3$:      ADDL3   R0, R2, BYTES_NEEDED                  ; 1730
                00000200     8F            51    D1 00040          CMPL    BYTES_NEEDED, #512                    ; 1740
                             25            15    00047             BLEQ    5$                                    ;
                     10      AE            51    D0 00049          MOVL    BYTES_NEEDED, UNWIND_VM_SIZE          ; 1743
                     08      AE            5B    D0 0004D          MOVL    CCB, UNWIND_CCB                       ; 1744
                                     0C    AE    9F 00051          PUSHAB  UNWIND_VM_ADDR                        ; 1745
                                     14    AE    9F 00054          PUSHAB  UNWIND_VM_SIZE                        ;
                0000000G     00            02    FB 00057          CALLS   #2, LIB$GET_VM                        ;
                             07            50    E8 0005E          BLBS    R0, 4$                                ;
                             7E      00G   8F    9A 00061          MOVZBL  #BAS$K_MAXMEMEXC, -(SP)               ;
                             6A            01    FB 00065          CALLS   #1, BAS$$STOP_IO                      ;
                             59       0C   AE    D0 00068 4$:      MOVL    UNWIND_VM_ADDR, CHARCONS              ; 1746
                             04            11    0006C             BRB     6$                                    ; 1740
                             59       2C   AE    9E 0006E 5$:      MOVAB   WORKSPACE, CHARCONS                   ; 1749
                             58       B8   AB    D0 00072 6$:      MOVL    -72(CCB), TEMP_CCB                    ; 1753
                             57       10   AC    D0 00076          MOVL    FORMAT, R7                            ; 1755
                                           03    14 0007A          BGTR    7$                                    ;
                                         0083    31 0007C          BRW     14$                                   ;
          7B          FE     AB            05    E1 0007F 7$:      BBC     #5, -2(CCB), 13$                      ; 1764
                             50       34   AB    9A 00084          MOVZBL  52(CCB), R0                           ; 1778
                     04      AE       30 BB40    9E 00088          MOVAB   @48(CCB)[R0], RDSC+4                  ;
                             56       08   AC    D0 0008E          MOVL    ELEM_SIZE, R6                         ; 1791
                             50       C8   A8    9E 00092          MOVAB   -56(TEMP_CCB), R0                     ; 1792
          02                 01            57    CF 00096          CASEL   R7, #1, #2                            ; 1785
        0038               0014          0006       0009A 8$:      .WORD   9$-8$,-                               ;
                                                                           10$-8$,-                              ;
                                                                           11$-8$                                ;
  96   AB          02        00            01    F0 000A0 9$:      INSV    #1, #0, #2, -106(CCB)                 ; 1790
                             6E            56    B0 000A6          MOVW    R6, RDSC                              ; 1791
                             60            56    C0 000A9          ADDL2   R6, (R0)                              ; 1792
                             2E            11    000AC             BRB     12$                                   ; 1785
  96   AB          02        00            02    F0 000AE 10$:     INSV    #2, #0, #2, -106(CCB)                 ; 1796
                   51        60            56    C1 000B4          ADDL3   R6, (R0), R1                          ; 1797
         7E        00        51            01    7A 000B8          EMUL    #1, R1, #0, -(SP)                     ; 1798
         51        51        8E            0E    7B 000BD          EDIV    #14, (SP)+, R1, R1                    ;
                   51        56            51    C3 000C2          SUBL3   R1, R6, R1                            ; 1797
                   6E        51            0E    A1 000C6          ADDW3   #14, R1, RDSC                         ;
                             51            6E    3C 000CA          MOVZWL  RDSC, R1                              ; 1799
                             60            51    C0 000CD          ADDL2   R1, (R0)                              ;
```

```
                                      0A  11 000D0         BRB     12$                                        ; 1785
               6E          56         02  A1 000D2 11$:    ADDW3   #2, R6, RDSC                               ; 1808
                      96   AB         03  88 000D6         BISB2   #3, -106(CCB)                              ; 1809
                                      60  D4 000DA         CLRL    (R0)                                       ; 1810
                      34   AB         6E  80 000DC 12$:    ADDB2   RDSC, 52(CCB)                              ; 1819
                      03   AE         01  90 000E0         MOVB    #1, RDSC+3                                 ; 1820
                           50    0C   AC  D0 000E4         MOVL    ELEM_ADR, R0                               ; 1821
     6E        20     04   B0         56  2C 000E8         MOVC5   R6, @4(R0), #32, RDSC, @RDSC+4             ;
                                 04   BE     000EE
                           03         57  D1 000F0         CMPL    R7, #3                                     ; 1822
                                      0A  12 000F3         BNEQ    13$                                        ;
               50          56    04   AE  C1 000F5         ADDL3   RDSC+4, R6, R0                             ; 1824
                           60  0A0D   8F  B0 000FA         MOVW    #2573, (R0)                                ;
                                    01DE  31 000FF 13$:    BRW     47$                                        ; 1826
                                      59  DD 00102 14$:    PUSHL   CHARCONS                                   ; 1863
                           52    04   AC  D0 00104         MOVL    ELEM_TYPE, R2                              ;
                                      52  DD 00108         PUSHL   R2                                         ;
               16          06         52  CF 0010A         CASEL   R2, #6, #22                                ; 1848
     002E    0032        0032       0032     0010E 15$:    .WORD   17$-15$,-                                  ;
     002E    002E        0032       0032     00116                 17$-15$,-                                  ;
     002E    002E        002E       0038     0011E                 17$-15$,-                                  ;
     0038    002E        002E       002E     00126                 16$-15$,-                                  ;
     002E    002E        002E       002E     0012E                 17$-15$,-                                  ;
             0032        0032       002E     00136                 16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   18$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   18$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   16$-15$,-                                  ;
                                                                   17$-15$,-                                  ;
                                                                   17$-15$                                    ;
                                      7E  D4 0013C 16$:    CLRL    -(SP)                                      ;
                                      0C  11 0013E         BRB     20$                                        ;
                           50    24   AE  9E 00140 17$:    MOVAB   D_VALUE, R0                                ;
                                      04  11 00144         BRB     19$                                        ;
                           50    1C   AE  9E 00146 18$:    MOVAB   DSC, R0                                    ;
                                      50  DD 0014A 19$:    PUSHL   R0                                         ;
                   0000V   CF         03  FB 0014C 20$:    CALLS   #3, GETFIELD                               ;
                           03         50  E9 00151         BLBC    R0, 21$                                    ;
                                    0087  31 00154         BRW     30$                                        ;
                                 FE   AB  95 00157 21$:    TSTB    -2(CCB)                                    ; 1870
                                      17  18 0015A         BGEQ    22$                                        ;
               12     A1   AB         04  E0 0015C         BBS     #4, -95(CCB), 22$                          ;
                           6E         02  D0 00161         MOVL    #2, TDSC                                   ; 1881
                      04   AE  FE94   CF  9E 00164         MOVAB   D_PROMPT, TDSC+4                           ; 1882
                                      5E  DD 0016A         PUSHL   SP                                         ; 1883
               00000000G   00         01  FB 0016C         CALLS   #1, BAS$OUT_T_DX_S                         ;
```

```
                                50      FF71    CB  9A 00173 22$:     MOVZBL  -143(CCB), R0                           ; 1897
                                50 00000000G0040    D0 00178          MOVL    BAS$$AA_REC_PR1-104[R0], R0             ;
                                   00000000G0040    16 00180          JSB     BAS$$AA_REC_PR1[R0]                     ;
                                06              50  E8 00187          BLBS    R0, 23$                                 ;
                                          34    AB  94 0018A          CLRB    52(CCB)                                 ; 1908
                                        0154    31 0018D              BRW     48$                                     ; 1909
                                          34    AB  94 00190 23$:     CLRB    52(CCB)                                 ; 1915
                                      0204      8F  BB 00193          PUSHR   #^M<R2,R9>                              ; 1939
                            16          06      52  CF 00197          CASEL   R2, #6, #22                             ; 1924
                    002E    0032        0032        0032    0019B 24$:    .WORD   26$-24$,-                           ;
                    002E    002E        0032        0032    001A3             26$-24$,-                               ;
                    002E    002E        002E        0038    001AB             26$-24$,-                               ;
                    0038    002E        002E        002E    001B3             25$-24$,-                               ;
                    002E    002E        002E        002E    001BB             26$-24$,-                               ;
                            0032        0032        002E    001C3             26$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              27$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              27$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              25$-24$,-                               ;
                                                                              26$-24$,-                               ;
                                                                              26$-24$                                 ;
                                          7E    D4  001C9 25$:    CLRL    -(SP)                                   ;
                                          0C    11  001CB         BRB     29$                                     ;
                                50      24  AE    9E  001CD 26$:    MOVAB   D_VALUE, R0                             ;
                                          04    11  001D1         BRB     28$                                     ;
                                50      1C  AE    9E  001D3 27$:    MOVAB   DSC, R0                                 ;
                                          50    DD  001D7 28$:    PUSHL   R0                                      ;
                                0000V     CF        03  FB  001D9 29$:    CALLS   #3, GETFIELD                    ;
                            16          06      52  CF 001DE 30$:    CASEL   R2, #6, #22                             ; 1947
                    00CE    003E        0037        0030    001E2 31$:    .WORD   32$-31$,-                           ;
                    00CE    00CE        0045        003E    001EA             33$-31$,-                               ;
                    00CE    00CE        00CE        005E    001F2             34$-31$,-                               ;
                    0084    00CE        00CE        00CE    001FA             45$-31$,-                               ;
                    00CE    00CE        00CE        00CE    00202             34$-31$,-                               ;
                            004F        0045        00CE    0020A             35$-31$,-                               ;
                                                                              45$-31$,-                               ;
                                                                              45$-31$,-                               ;
                                                                              38$-31$,-                               ;
                                                                              45$-31$,-                               ;
                                                                              45$-31$,-                               ;
                                                                              45$-31$,-                               ;
                                                                              45$-31$,-                               ;
                                                                              45$-31$,-                               ;
                                                                              45$-31$,-                               ;
                                                                              39$-31$,-                               ;
                                                                              45$-31$,-                               ;
```

```
                                                                 45$-31$,-                              ;
                                                                 45$-31$,-                              ;
                                                                 45$-31$,-                              ;
                                                                 45$-31$,-                              ;
                                                                 35$-31$,-                              ;
                                                                 36$-31$                                ;
                                   2C   11 00210         BRB     37$                                    ;
          0C    BC         1C      AE   90 00212 32$:    MOVB    D_VALUE, @ELEM_ADR                     ; 1962
                                   25   11 00217         BRB     37$                                    ; 1947
          0C    BC         1C      AE   B0 00219 33$:    MOVW    D_VALUE, @ELEM_ADR                     ; 1971
                                   1E   11 0021E         BRB     37$                                    ; 1947
          0C    BC         1C      AE   D0 00220 34$:    MOVL    D_VALUE, @ELEM_ADR                     ; 1977
                                   17   11 00225         BRB     37$                                    ;
                50         0C      AC   D0 00227 35$:    MOVL    ELEM_ADR, R0                           ; 1983
                60         1C      AE   7D 0022B         MOVQ    D_VALUE, (R0)                          ;
                                   7F   11 0022F         BRB     45$                                    ; 1947
                50         0C      AC   D0 00231 36$:    MOVL    ELEM_ADR, R0                           ; 1991
                60         1C      AE   7D 00235         MOVQ    D_VALUE, (R0)                          ;
          08    A0         24      AE   7D 00239         MOVQ    D_VALUE+8, 8(R0)                       ; 1993
                                   70   11 0023E 37$:    BRB     45$                                    ; 1947
          18    AE                 59   D0 00240 38$:    MOVL    CHARCONS, DSC+4                        ; 2001
          16    AE       010E      8F   B0 00244         MOVW    #270, DSC+2                            ; 2003
                           14      AE   9F 0024A         PUSHAB  DSC                                    ; 2005
                           0C      AC   DD 0024D         PUSHL   ELEM_ADR                               ;
   00000000G    00                 02   FB 00250         CALLS   #2, STR$COPY_DX                        ;
                1A         18      BE   91 00257         CMPB    @DSC+4, #26                            ; 2006
                                   53   12 0025B         BNEQ    45$                                    ;
                           34      AB   94 0025D         CLRB    52(CCB)                                ; 2016
                7E        00G      8F   9A 00260         MOVZBL  #BAS$K_ENDFILDEV, -(SP)                ; 2017
                                   47   11 00264         BRB     44$                                    ;
          18    AE                 59   D0 00266 39$:    MOVL    CHARCONS, DSC+4                        ; 2033
          16    AE       010E      8F   B0 0026A         MOVW    #270, DSC+2                            ; 2035
                50                 5D   D0 00270         MOVL    FP, FMP                                ; 2041
                50         0C      A0   D0 00273 40$:    MOVL    12(FMP), FMP                           ; 2045
                51  00000000G      00   9E 00277         MOVAB   BAS$HANDLER, R1                        ; 2047
                51                 60   D1 0027E         CMPL    (FMP), R1                              ;
                                   04   13 00281         BEQL    41$                                    ;
                                   50   D5 00283         TSTL    FMP                                    ; 2048
                                   EC   12 00285         BNEQ    40$                                    ;
                                   50   D5 00287 41$:    TSTL    FMP                                    ; 2050
                                   09   13 00289         BEQL    42$                                    ;
  04      E6    A0                 09   E1 0028B         BBC     #9, -26(FMP), 42$                      ;
                                   50   D4 00290         CLRL    FLAGS                                  ; 2052
                                   03   11 00292         BRB     43$                                    ;
                50                 08   D0 00294 42$:    MOVL    #8, FLAGS                              ; 2054
                                   50   DD 00297 43$:    PUSHL   FLAGS                                  ; 2056
                           0C      AC   DD 00299         PUSHL   ELEM_ADR                               ;
                           1C      AE   9F 0029C         PUSHAB  DSC                                    ;
   00000000G    00                 03   FB 0029F         CALLS   #3, BAS$CVT_T_P                        ;
                07                 50   E8 002A6         BLBS    STATUS, 45$                            ; 2057
                7E        00G      8F   9A 002A9         MOVZBL  #BAS$K_DATFORERR, -(SP)                ;
                6A                 01   FB 002AD 44$:    CALLS   #1, BAS$$STOP_IO                       ;
                           34      AB   94 002B0 45$:    CLRB    52(CCB)                                ; 2060
                1B         0C      AB   B1 002B3         CMPW    12(CCB), #27                           ; 2061
                                   04   13 002B7         BEQL    46$                                    ;
          FE    A8                 04   8A 002B9         BICB2   #4, -2(TEMP_CCB)                       ;
                50         2C      AE   9E 002BD 46$:    MOVAB   WORKSPACE, R0                          ; 2065
```

```
                        50              59  D1 002C1            CMPL    CHARCONS, R0                                    ;
                                        1A  13 002C4            BEQL    47$                                             ;
                                  0C    AE  9F 002C6            PUSHAB  UNWIND_VM_ADDR                                  ; 2068
                                  14    AE  9F 002C9            PUSHAB  UNWIND_VM_SIZE                                  ;
                 00000000G      00      02  FB 002CC            CALLS   #2, LIB$FREE_VM                                 ;
                                0A      50  E8 002D3            BLBS    R0, 47$                                         ;
                                  10    AE  D4 002D6            CLRL    UNWIND_VM_SIZE                                  ; 2071
                                7E      00G 8F 9A 002D9         MOVZBL  #BAS$K_PROLOSSOR, -(SP)                         ; 2072
                                6A      01  FB 002DD            CALLS   #1, BAS$$STOP_IO                                ;
                                50      01  D0 002E0 47$:      MOVL    #1, R0                                          ; 2075
                                            04 002E3            RET                                                     ;
                                        50  D4 002E4 48$:      CLRL    R0                                              ; 2076
                                            04 002E6            RET                                                     ;
                                      0000  002E7 49$:      .WORD   Save nothing                                    ; 1689
                                50      08  AC  D0 002E9         MOVL    8(AP), R0                                       ;
                                50      04  A0  D0 002ED         MOVL    4(R0), R0                                       ;
                                      FDDC  C0  9F 002F1         PUSHAB  UNWIND_CCB                                      ;
                                      FDE0  C0  9F 002F5         PUSHAB  UNWIND_VM_ADDR                                  ;
                                      FDE4  C0  9F 002F9         PUSHAB  UNWIND_VM_SIZE                                  ;
                                            03  DD 002FD         PUSHL   #3                                              ;
                                            5E  DD 002FF         PUSHL   SP                                              ;
                                7E      04  AC  7D 00301         MOVQ    4(AP), -(SP)                                    ;
                         0000V  CF              03  FB 00305         CALLS   #3, UDF_RL1_HANDLER                             ;
                                                04 0030A         RET                                                     ;

; Routine Size:  779 bytes,    Routine Base:  _BAS$CODE + 003C

;  784          2077  1
```

```
;   786          2078  1 ROUTINE UDF_RL1_HANDLER (                       !Handeler for bas$udf_rl1
;   787          2079  1         SIG                                     !Signal vector
;   788          2080  1         ,MECH                                   !Mechanism vector
;   789          2081  1         ,ENBL                                   !Enable vector
;   790          2082  1         ) =
;   791          2083  1
;   792          2084  1 !++
;   793          2085  1 ! FUNCTIONAL DESCRIPTION:
;   794          2086  1 !
;   795          2087  1 !       If we are unwinding and we have given the parsing space VM then
;   796          2088  1 !       free this VM.
;   797          2089  1 !
;   798          2090  1 ! FORMAL PARAMETERS:
;   799          2091  1 !
;   800          2092  1 !       SIG.rl.ra       A counted vector of parameters from LIB$SIGNAL/STOP
;   801          2093  1 !       MECH.rl.ra      A counted vector of info from chf
;   802          2094  1 !       ENBL.rl.ra      A counted vector of ENABLE argument addresses.
;   803          2095  1 !
;   804          2096  1 ! IMPLICIT INPUTS
;   805          2097  1 !
;   806          2098  1 !       NONE
;   807          2099  1 !
;   808          2100  1 ! IMPLICIT OUTPUTS
;   809          2101  1 !
;   810          2102  1 !       NONE
;   811          2103  1 !
;   812          2104  1 ! COMPLETION CODES
;   813          2105  1 !
;   814          2106  1 !       Always SS$_RESIGNAL, which is ignored when unwinding.
;   815          2107  1 !
;   816          2108  1 ! SIDE EFFECTS
;   817          2109  1 !
;   818          2110  1 !       NONE
;   819          2111  1 !
;   820          2112  1 !--
;   821          2113  1
;   822          2114  2     BEGIN
;   823          2115  2
;   824          2116  2     MAP
;   825          2117  2         SIG : REF VECTOR,
;   826          2118  2         MECH: REF VECTOR,
;   827          2119  2         ENBL: REF VECTOR;
;   828          2120  2
;   829          2121  2     GLOBAL REGISTER CCB = K_CCB_REG : REF BLOCK [,BYTE];
;   830          2122  2
;   831          2123  2     CCB = ..ENBL [3];
;   832          2124  2 !+
;   833          2125  2 ! If we are unwinding and have allocated VM then free it.
;   834          2126  2 !-
;   835          2127  3     IF (LIB$MATCH_COND ( SIG [1] , %REF(SS$_UNWIND) ) AND ( ..ENBL [1] GTRU 0 ))
;   836          2128  2     THEN
;   837          2129  2         IF NOT LIB$FREE_VM ( .ENBL [1] , .ENBL [2] )
;   838          2130  2         THEN BAS$$STOP_IO ( BAS$K_PROLOSSOR );
;   839          2131  2     RETURN (SS$_RESIGNAL);
;   840          2132  2
;   841          2133  1     END;
```

```
                                        0804 00000 UDF_RL1_HANDLER:
                                                   .WORD   Save R2,R11                          ; 2078
                          52      0C    AC   D0 00002          MOVL    ENBL, R2                 ; 2123
                          5B      0C    B2   D0 00006          MOVL    @12(R2), CCB             ;
                          7E    0920    8F   3C 0000A          MOVZWL  #2336, -(SP)             ; 2127
                                        5E   DD 0000F          PUSHL   SP                       ;
                    7E      04    AC        04 C1 00011       ADDL3   #4, SIG, -(SP)           ;
                00000000G   00          02   FB 00016          CALLS   #2, LIB$MATCH_COND       ;
                          1E            50   E9 0001D          BLBC    R0, 1$                   ;
                                  04    B2   D5 00020          TSTL    @4(R2)                   ;
                                        19   13 00023          BEQL    1$                       ;
                          7E      04    A2   7D 00025          MOVQ    4(R2), -(SP)             ; 2129
                00000000G   00          02   FB 00029          CALLS   #2, LIB$FREE_VM          ;
                          0B            50   E8 00030          BLBS    R0, 1$                   ;
                          7E      00G   8F   9A 00033          MOVZBL  #BAS$K_PROLOSSOR, -(SP)  ; 2130
                00000000G   00          01   FB 00037          CALLS   #1, BAS$$STOP_IO         ;
                          50    0918    8F   3C 0003E  1$:     MOVZWL  #2328, R0                ; 2131
                                        04 00043               RET                              ; 2133

; Routine Size:  68 bytes,    Routine Base:  _BAS$CODE + 0347
```

```
;   843          2134  1 GLOBAL ROUTINE BAS$UDF_RL9
;   844          2135  1         : JSB_UDF9 NOVALUE =
;   845          2136  1
;   846          2137  1 !++
;   847          2138  1 ! FUNCTIONAL DESCRIPTION:
;   848          2139  1 !
;   849          2140  1 !       List directed input UDF termination.
;   850          2141  1 !
;   851          2142  1 ! FORMAL PARAMETERS:
;   852          2143  1 !
;   853          2144  1 !       NONE
;   854          2145  1 !
;   855          2146  1 ! IMPLICIT INPUTS:
;   856          2147  1 !
;   857          2148  1 !       NONE
;   858          2149  1 !
;   859          2150  1 ! IMPLICIT OUTPUTS:
;   860          2151  1 !
;   861          2152  1 !       NONE
;   862          2153  1 !
;   863          2154  1 ! ROUTINE VALUE:
;   864          2155  1 ! COMPLETION CODES:
;   865          2156  1 !
;   866          2157  1 !       NONE
;   867          2158  1 !
;   868          2159  1 ! SIDE EFFECTS:
;   869          2160  1 !
;   870          2161  1 !       NONE
;   871          2162  1 !
;   872          2163  1 !--
;   873          2164  1
;   874          2165  2     BEGIN
;   875          2166  2
;   876          2167  2     RETURN;
;   877          2168  1     END;


                                                    05 00000 BAS$UDF_RL9::
                                                             RSB                                    ; 2168

; Routine Size:  1 bytes,    Routine Base:  _BAS$CODE + 038B
```

```
;  879          2169  1 ROUTINE GETFIELD (
;  880          2170  1         ELEM,
;  881          2171  1         ELEM_TYPE,
;  882          2172  1         WORK_STR
;  883          2173  1         ) :CALL_CCB =
;  884          2174  1
;  885          2175  1 !++
;  886          2176  1 ! FUNCTIONAL DESCRIPTION:
;  887          2177  1 !
;  888          2178  1 !       Parse out the next input data field based on the field terminators
;  889          2179  1 !       appropriate for the data type.  Return the field with tabs and spaces
;  890          2180  1 !       stripped out in the area supplied by the calling routine.
;  891          2181  1 !       A one is returned if a field was found.  A zero is returned if an <eol>
;  892          2182  1 !       is encountered before a field is found.
;  893          2183  1 !
;  894          2184  1 ! FORMAL PARAMETERS:
;  895          2185  1 !
;  896          2186  1 !       ELEM_TYPE.rlu.v         Type of element from list
;  897          2187  1 !       ELEM.wz.r               Pointer of where to return the value
;  898          2188  1 !                               May be a reference to a quadword or a descriptor
;  899          2189  1 !       WORK_STR.wt.rs          Work string for parsing input string and resulting
;  900          2190  1 !                               string for type text.
;  901          2191  1 !
;  902          2192  1 ! IMPLICIT INPUTS:
;  903          2193  1 !
;  904          2194  1 !       LUB$A_BUF_PTR           current location in the buffer
;  905          2195  1 !       LUB$A_BUF_END           pointer to last byte of buffer + 1
;  906          2196  1 !       RAB$W_RSZ               buffer size
;  907          2197  1 !       RAB$W_STV0              first word of STV field
;  908          2198  1 !       RAB$W_STV2              second word of STV field
;  909          2199  1 !       ISB$B_STTM_TYPE         I/O statement type in ISB
;  910          2200  1 !
;  911          2201  1 ! IMPLICIT OUTPUTS:
;  912          2202  1 !
;  913          2203  1 !       LUB$A_BUF_PTR           Pointer to next byte in user buffer
;  914          2204  1 !       ISB$B_ERR_NO            first error found processing an I/O stmt.
;  915          2205  1 !
;  916          2206  1 ! ROUTINE VALUE:
;  917          2207  1 !
;  918          2208  1 !       1 = a data field was found
;  919          2209  1 !       0 = a data field was not found
;  920          2210  1 !
;  921          2211  1 ! COMPLETION CODES:
;  922          2212  1 !
;  923          2213  1 !       NONE
;  924          2214  1 !
;  925          2215  1 ! SIDE EFFECTS:
;  926          2216  1 !
;  927          2217  1 !       NONE
;  928          2218  1 !
;  929          2219  1 !--
;  930          2220  1
;  931          2221  1     !+
;  932          2222  1     ! Note: There are 3 exit points from this routine; not the best structure
;  933          2223  1     ! but that's the way it is.
;  934          2224  1     !-
;  935          2225  1
```

```
;  936          2226  2      BEGIN
;  937          2227  2
;  938          2228  2      MAP
;  939          2229  2          WORK_STR: REF VECTOR [K_WORK_STR_LEN, BYTE],    ! work area
;  940          2230  2          ELEM: REF VECTOR [2];                           ! default ELEM to numeric
;  941          2231  2
;  942          2232  2      LOCAL
;  943          2233  2          DSC: BLOCK [8, BYTE],                           ! working desc. for parsing the input stream
;  944          2234  2          MASK,                                           ! mask value for SCANC
;  945          2235  2          RET_VAL,                                        ! value to return to caller
;  946          2236  2          LEN,                                            ! length of input record remaining to be scanned
;  947          2237  2          SCAN_VAL,                                       ! return value from SCANC
;  948          2238  2          PTRS,                                           ! char pointer for source string
;  949          2239  2          PTRD;                                           ! char pointer for destination string
;  950          2240  2
;  951          2241  2          !+
;  952          2242  2          ! mask values for the SCANC to stop on different characters
;  953          2243  2          !-
;  954          2244  2
;  955          2245  2      LITERAL
;  956          2246  2          K_COMMA = %X'01',
;  957          2247  2          K_SEMI = %X'02',
;  958          2248  2          K_SGL_QUOTE = %X'04',
;  959          2249  2          K_DBL_QUOTE = %X'08',
;  960          2250  2          K_TAB_SPACE = %X'10',
;  961          2251  2          K_NULL = %X'20',
;  962          2252  2          K_CHAR = %X'40',
;  963          2253  2          K_NONE = %X'00',
;  964          2254  2  !+
;  965          2255  2  ! The element size of a longword integer
;  966          2256  2  !-
;  967          2257  2          K_INT_SIZ = 4,
;  968          2258  2  !+
;  969          2259  2  ! The flags for floating and integer input
;  970          2260  2  !-
;  971          2261  2          K_INT_FLAGS = 5,
;  972          2262  2          K_FLT_F_FLAGS = 123,
;  973          2263  2          K_FLT_D_FLAGS = 115;
;  974          2264  2
;  975          2265  2      BIND
;  976          2266  2          TABLE = UPLIT BYTE (
;  977          2267  2              %X'20', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  978          2268  2                  %X'10', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', ! column 0
;  979          2269  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  980          2270  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', ! column 1
;  981          2271  2              %X'10', %X'40', %X'48', %X'40', %X'40', %X'40', %X'40', %X'44', %X'40',
;  982          2272  2                  %X'40', %X'40', %X'40', %X'41', %X'40', %X'40', %X'40', %X'40', ! column 2
;  983          2273  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  984          2274  2                  %X'40', %X'40', %X'42', %X'40', %X'40', %X'40', %X'40', %X'40', ! column 3
;  985          2275  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  986          2276  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', ! column 4
;  987          2277  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  988          2278  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', ! column 5
;  989          2279  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  990          2280  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', ! column 6
;  991          2281  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  992          2282  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', ! column 7
```

```
;  993          2283  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  994          2284  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40' ! column 8
;  995          2285  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  996          2286  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40' ! column 9
;  997          2287  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
;  998          2288  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40' ! column 10
;  999          2289  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
; 1000          2290  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40' ! column 11
; 1001          2291  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
; 1002          2292  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40' ! column 12
; 1003          2293  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
; 1004          2294  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40' ! column 13
; 1005          2295  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
; 1006          2296  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40' ! column 14
; 1007          2297  2              %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40',
; 1008          2298  2                  %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40', %X'40' ! column 15
; 1009          2299  2          ): VECTOR[256, BYTE];
; 1010          2300
; 1011          2301  2      EXTERNAL REGISTER
; 1012          2302  2          CCB: REF BLOCK [, BYTE];
; 1013          2303
; 1014          2304  2      !+
; 1015          2305  2      ! Initialize the default null string (zero length)
; 1016          2306  2      !-
; 1017          2307
; 1018          2308  2      DSC[DSC$W_LENGTH] = 0;
; 1019          2309
; 1020          2310  2      !+
; 1021          2311  2      ! Check to see if there is any more data in the record.
; 1022          2312  2      ! If there is no more data (BUF_PTR GEQA BUF_END) then return a failure
; 1023          2313  2      ! status.  Otherwise, increment BUF_PTR.
; 1024          2314  2      !-
; 1025          2315
; 1026          2316  2      IF .CCB[LUB$A_BUF_PTR] GEQA .CCB[LUB$A_BUF_END]
; 1027          2317  2      THEN
; 1028          2318  2          RETURN 0
; 1029          2319  2      ELSE
; 1030          2320  2          CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_PTR] + 1;
; 1031          2321
; 1032          2322  2      !+
; 1033          2323  2      ! Check for the buffer pointer equal to the end of the buffer (return default).
; 1034          2324  2      ! If the statement type is INPUT LINE, we will do all of the other processing.
; 1035          2325  2      ! For ANSI INPUT, no defaults should be applied.  Signal the 'too little data'
; 1036          2326  2      ! error for ANSI.
; 1037          2327  2      !-
; 1038          2328
; 1039          2329  3      IF (.CCB [LUB$A_BUF_PTR] EQLA .CCB [LUB$A_BUF_END])
; 1040          2330  2          AND .CCB [LOB$V_ANSI]
; 1041          2331  2      THEN
; 1042          2332  2          BAS$$SIGNAL_IO (BAS$K_TOOLITDAT);
; 1043          2333
; 1044          2334  3      IF (.CCB[LUB$A_BUF_PTR] EQLA .CCB[LUB$A_BUF_END])
; 1045          2335  3       AND (.CCB [ISB$B_STTM_TYPE] NEQ ISB$K_ST_TY_INL)
; 1046          2336  2      THEN
; 1047          2337
; 1048          2338  2          !+
; 1049          2339  2          ! Return a zero or a null string as a default value
```

```
;  1050          2340  2          !-
;  1051          2341  2
;  1052          2342  2          BEGIN
;  1053          2343  2          CASE .ELEM_TYPE
;  1054          2344  2          FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
;  1055          2345  2          SET
;  1056          2346  2          [INRANGE, OUTRANGE]:
;  1057          2347  2              !+
;  1058          2348  2              ! Data types not yet supported
;  1059          2349  2              !-
;  1060          2350  2              ELEM[0] = 0;
;  1061          2351  2          [DSC$K_DTYPE_B, DSC$K_DTYPE_W, DSC$K_DTYPE_L, DSC$K_DTYPE_F]:
;  1062          2352  2
;  1063          2353  2              !+
;  1064          2354  2              ! Data type integer
;  1065          2355  2              !-
;  1066          2356  2
;  1067          2357  2              ELEM[0] = 0;
;  1068          2358  2          [DSC$K_DTYPE_D, DSC$K_DTYPE_G]:
;  1069          2359  2
;  1070          2360  2              !+
;  1071          2361  2              ! Data type double precision or g float
;  1072          2362  2              !-
;  1073          2363  2
;  1074          2364  2              BEGIN
;  1075          2365  2              ELEM[0] = 0;
;  1076          2366  2              ELEM[1] = 0;
;  1077          2367  2              END;
;  1078          2368  2          [DSC$K_DTYPE_H]:
;  1079          2369  2
;  1080          2370  3              !+
;  1081          2371  3              ! Data type h float
;  1082          2372  3              !-
;  1083          2373  4              BEGIN
;  1084          2374  4              ELEM[0] = 0;
;  1085          2375  4              ELEM[1] = 0;
;  1086          2376  4              ELEM[2] = 0;
;  1087          2377  4              ELEM[3] = 0;
;  1088          2378  3              END;
;  1089          2379  3          [DSC$K_DTYPE_T, DSC$K_DTYPE_P]:
;  1090          2380  3
;  1091          2381  3              !+
;  1092          2382  3              ! Data type text or packed decimal string
;  1093          2383  3              !-
;  1094          2384  4              BEGIN
;  1095          2385  4              MAP
;  1096          2386  4                  ELEM: REF BLOCK [8, BYTE];
;  1097          2387  4              ELEM[DSC$W_LENGTH] = 0;
;  1098          2388  3              END;
;  1099          2389  3          TES;
;  1100          2390  3          RETURN 1;
;  1101          2391  2          END;
;  1102          2392  2
;  1103          2393  2      !+
;  1104          2394  2      ! Set up the mask for the scan.  Make any special adjustments to the buffer
;  1105          2395  2      ! pointer that are necessary for type character string.
;  1106          2396  2      !-
```

```
;  1107          2397  2
;  1108          2398  2        DSC[DSC$A_POINTER] = WORK_STR[0];
;  1109          2399  2        CASE .ELEM_TYPE
;  1110          2400  2        FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
;  1111          2401  2        SET
;  1112          2402  2        [INRANGE, OUTRANGE]:
;  1113          2403  2            !+
;  1114          2404  2            ! Data types which are not supported yet
;  1115          2405  2            !-
;  1116          2406  2            0;
;  1117          2407  2        [DSC$K_DTYPE_B, DSC$K_DTYPE_W, DSC$K_DTYPE_L, DSC$K_DTYPE_F, DSC$K_DTYPE_D,
;  1118          2408  2         DSC$K_DTYPE_G, DSC$K_DTYPE_H, DSC$K_DTYPE_P]:
;  1119          2409  2            MASK = K_COMMA OR K_TAB_SPACE OR K_NULL;
;  1120          2410  2        [DSC$K_DTYPE_T]:
;  1121          2411  2
;  1122          2412  2            !+
;  1123          2413  2            ! First check for INPUT LINE, MAT LINPUT, or LINPUT.  They return the whole line regardless
;  1124          2414  2            ! of the contents.  Remove all leading tabs and spaces.  Next check for
;  1125          2415  2            ! quotes (single or double).  They return
;  1126          2416  2            ! everything up to the matched quote.  The quotes themselves are not returned
;  1127          2417  2            ! and the first one is stripped off by incrementing the buffer pointer.
;  1128          2418  2            ! Otherwise, a field is delimited by a comma or <eol>.
;  1129          2419  2            ! Trailing spaces and tabs are stripped off unquoted strings at great
;  1130          2420  2            ! pain.
;  1131          2421  2            !-
;  1132          2422  2
;  1133          2423  2            IF .CCB[ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_LIN
;  1134          2424  2             OR .CCB[ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_INL
;  1135          2425  2             OR .CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_MLI
;  1136          2426  2            THEN
;  1137          2427  2                MASK = K_NONE
;  1138          2428  2            ELSE
;  1139          2429  2                BEGIN
;  1140          2430  2
;  1141          2431  2                !+
;  1142          2432  2                ! Strip off the leading tabs, nulls, and spaces.  If this results
;  1143          2433  2                ! in a zero length string then return the null string.
;  1144          2434  2                !-
;  1145          2435  2
;  1146          2436  2                WHILE (.(.CCB [LUB$A_BUF_PTR])<0,8,0> EQL %C' '
;  1147          2437  2                 OR .(.CCB [LUB$A_BUF_PTR])<0,8,0> EQL %C'	'
;  1148          2438  2                 OR .(.CCB [LUB$A_BUF_PTR])<0,8,0> EQL %X'00')
;  1149          2439  2                 AND .CCB [LUB$A_BUF_PTR] LSS .CCB [LUB$A_BUF_END]
;  1150          2440  2                DO
;  1151          2441  2                    CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_PTR] + 1;
;  1152          2442  2                IF .CCB[LUB$A_BUF_PTR] GEQ .CCB[LUB$A_BUF_END]
;  1153          2443  2                 OR .(.CCB[LUB$A_BUF_PTR])<0,8,0> EQL %C','
;  1154          2444  2                THEN
;  1155          2445  2                    BEGIN
;  1156          2446  2                    MAP
;  1157          2447  2                        ELEM: REF BLOCK [8, BYTE];
;  1158          2448  2                    ELEM[DSC$W_LENGTH] = 0;
;  1159          2449  2                    RETURN 1;
;  1160          2450  2                    END;
;  1161          2451  2                IF .(.CCB[LUB$A_BUF_PTR])<0, 8> EQL %C''''
;  1162          2452  2                THEN
;  1163          2453  2                    BEGIN
```

```
;  1164          2454  4                MASK = K_SGL_QUOTE;
;  1165          2455  4                CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_PTR] + 1;
;  1166          2456  4                END
;  1167          2457  4            ELSE
;  1168          2458  4                IF .(.CCB[LUB$A_BUF_PTR])<0, 8> EQL %C'"'
;  1169          2459  4                THEN
;  1170          2460  4                    BEGIN
;  1171          2461  4                    MASK = K_DBL_QUOTE;
;  1172          2462  4                    CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_PTR] + 1;
;  1173          2463  4                    END
;  1174          2464  4                ELSE
;  1175          2465  4                    MASK = K_COMMA;
;  1176          2466  4                END;
;  1177          2467  3        TES;
;  1178          2468  3
;  1179          2469  3        !+
;  1180          2470  3        ! Point the character pointer to the start of the field.
;  1181          2471  3        !-
;  1182          2472  3
;  1183          2473  3        PTRS = CH$PTR(.CCB[LUB$A_BUF_PTR]);
;  1184          2474  3        PTRD = CH$PTR(.DSC[DSC$A_POINTER]);
;  1185          2475  3        LEN = .CCB[LUB$A_BUF_END] - .CCB[LUB$A_BUF_PTR];
;  1186          2476  3
;  1187          2477  3        !+
;  1188          2478  3        ! Based on the data type, scan the input data string for an element
;  1189          2479  3        !-
;  1190          2480  3
;  1191          2481  3        WHILE 1 DO
;  1192          2482  3            BEGIN
;  1193          2483  3            LITERAL
;  1194          2484  3                K_DECIMAL_PT = %X'2E';
;  1195          2485  3            LOCAL
;  1196          2486  3                TEMP_LEN;                       !Used to allow > 64kb data
;  1197          2487  3            TEMP_LEN = (IF .LEN GEQU 65536 THEN 65535 ELSE .LEN);
;  1198          2488  3            SCAN_VAL = SCANC(TEMP_LEN, .CCB[LUB$A_BUF_PTR], TABLE, MASK);
;  1199          2489  3            IF .SCAN_VAL NEQ 0
;  1200          2490  3            THEN
;  1201          2491  3                CASE .ELEM_TYPE
;  1202          2492  3                FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
;  1203          2493  3                SET
;  1204          2494  3                [DSC$K_DTYPE_B, DSC$K_DTYPE_W, DSC$K_DTYPE_L, DSC$K_DTYPE_F,
;  1205          2495  3                 DSC$K_DTYPE_D, DSC$K_DTYPE_G, DSC$K_DTYPE_H, DSC$K_DTYPE_P]:
;  1206          2496  4                    BEGIN
;  1207          2497  4                    CH$MOVE (.SCAN_VAL-.CCB[LUB$A_BUF_PTR], .PTRS, .PTRD);
;  1208          2498  5                    IF      (.(.SCAN_VAL)<0, 8> EQL K_TAB)
;  1209          2499  5                        OR (.(.SCAN_VAL)<0, 8> EQL K_SP)
;  1210          2500  5                        OR (.(.SCAN_VAL)<0, 8> EQL %X'00')
;  1211          2501  4                    THEN
;  1212          2502  4
;  1213          2503  4                        !+
;  1214          2504  4                        ! A tab, null, or a space has been found in a numeric field
;  1215          2505  4                        ! Strip it out.
;  1216          2506  4                        ! Also strips out decimal points for packed decimal.
;  1217          2507  4                        !-
;  1218          2508  4
;  1219          2509  5                        BEGIN
;  1220          2510  5                        DSC[DSC$W_LENGTH] = .DSC[DSC$W_LENGTH] + (.SCAN_VAL - .CCB[LUB$A_BUF_PTR]);
```

```
;  1221          2511  5                    PTRS = CH$PLUS(.PTRS, .SCAN_VAL-.CCB[LUB$A_BUF_PTR] + 1);
;  1222          2512  5                    PTRD = CH$PLUS(.PTRD, .SCAN_VAL - .CCB[LUB$A_BUF_PTR]);
;  1223          2513  5                    LEN = .LEN - (.SCAN_VAL - .CCB[LUB$A_BUF_PTR]) - 1;
;  1224          2514  5                    CCB[LUB$A_BUF_PTR] = .SCAN_VAL + 1;
;  1225          2515  5                    END
;  1226          2516  4                ELSE
;  1227          2517  5                    BEGIN
;  1228          2518  5                    IF .SCAN_VAL EQLU .CCB[LUB$A_BUF_PTR]
;  1229          2519  5                    THEN
;  1230          2520  5
;  1231          2521  5                        !+
;  1232          2522  5                        ! An element separator was encountered as the next character;
;  1233          2523  5                        ! return the proper default value or the data scanned so far.
;  1234          2524  5                        !-
;  1235          2525  6                        BEGIN
;  1236          2526  6                        RET_VAL = 1;
;  1237          2527  6                        EXITLOOP;
;  1238          2528  5                        END;
;  1239          2529  5                    DSC[DSC$W_LENGTH] = .DSC[DSC$W_LENGTH] + .SCAN_VAL - .CCB[LUB$A_BUF_PTR];
;  1240          2530  5                    LEN = .LEN - (.SCAN_VAL - .CCB[LUB$A_BUF_PTR]) - 1;
;  1241          2531  5                    CCB[LUB$A_BUF_PTR] = .SCAN_VAL;
;  1242          2532  5                    RET_VAL = 1;
;  1243          2533  5                    EXITLOOP;
;  1244          2534  4                    END;
;  1245          2535  3                END;
;  1246          2536  3
;  1247          2537  3            [DSC$K_DTYPE_T]:
;  1248          2538  3                !+
;  1249          2539  3                ! Type text
;  1250          2540  3                ! Update the length so far, move the substring found, and
;  1251          2541  3                ! check for a delimiting quote if necessary.
;  1252          2542  3                !-
;  1253          2543  3
;  1254          2544  4                BEGIN
;  1255          2545  4                LOCAL
;  1256          2546  4                    A_HIGH_MARK;                ! High water mark of SCAN
;  1257          2547  4
;  1258          2548  4                !+
;  1259          2549  4                ! Strip off trailing spaces, nulls, and tabs if unquoted string
;  1260          2550  4                !-
;  1261          2551  4
;  1262          2552  4                A_HIGH_MARK = .SCAN_VAL;
;  1263          2553  4                IF .MASK EQL K_COMMA
;  1264          2554  4                THEN
;  1265          2555  4                    WHILE .(.SCAN_VAL - 1)<0,8,0> EQL %C' '
;  1266          2556  4                     OR .(.SCAN_VAL - 1)<0,8,0> EQL %C'	'
;  1267          2557  4                     OR .(.SCAN_VAL - 1)<0,8,0> EQL %X'00'
;  1268          2558  4                    DO
;  1269          2559  4                        SCAN_VAL = .SCAN_VAL - 1;
;  1270          2560  4
;  1271          2561  4                DSC[DSC$W_LENGTH] = .SCAN_VAL - .CCB[LUB$A_BUF_PTR];
;  1272          2562  4                CH$MOVE (.SCAN_VAL - .CCB[LUB$A_BUF_PTR], .PTRS, .PTRD);
;  1273          2563  4
;  1274          2564  4                !+
;  1275          2565  4                ! increment the buffer pointer if a delimiting quote is present
;  1276          2566  4                !-
;  1277          2567  4
```

```
;  1278          2568  4                    CCB[LUB$A_BUF_PTR] = .A_HIGH_MARK;
;  1279          2569  4                    IF .(.A_HIGH_MARK)<0, 8> EQL %C''' OR .(.A_HIGH_MARK)<0, 8> EQL %C'"'
;  1280          2570  4                    THEN
;  1281          2571  5                        BEGIN
;  1282          2572  5                        LOCAL
;  1283          2573  5                            T_RET_VAL,                  ! temp return value from SCANC
;  1284          2574  5                                                        ! looking for delimiting comma
;  1285          2575  5                            REM_LENGTH;                 ! Length remaining in the buffer
;  1286          2576  5                        CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_PTR] + 1;
;  1287          2577  5                !+
;  1288          2578  5                ! Scan for a comma, another character or the end-of-record following this quoted string.
;  1289          2579  5                ! Set BUF_PTR to the address that the scan returns.  If there is a comma,
;  1290          2580  5                ! then it will be pointing at the comma.
;  1291          2581  5                ! If there is a character other than space, tab or null following qoute, signal.
;  1292          2582  5                !-
;  1293          2583  5                        MASK = K_COMMA OR K_CHAR;
;  1294          2584  5                        REM_LENGTH = .LEN - .DSC [DSC$W_LENGTH] - 1;
;  1295          2585  5                        REM_LENGTH = (IF .REM_LENGTH GEQU 65536 THEN 65535 ELSE .REM_LENGTH);
;  1296          2586  5                        T_RET_VAL = SCANC(REM_LENGTH, .CCB [LUB$A_BUF_PTR],
;  1297          2587  5                            TABLE, MASK);
;  1298          2588  5                        CCB [LUB$A_BUF_PTR] = (IF .T_RET_VAL EQL 0 THEN .CCB [LUB$A_BUF_END] + 1 ELSE .T_RET_VAL
;  1299          2589  5                        IF (.T_RET_VAL NEQ 0) AND
;  1300          2590  6                           (.(.T_RET_VAL)< 0 , 8 > NEQ %C',')
;  1301          2591  5                        THEN BAS$$STOP_IO ( BAS$K_DATFORERR );
;  1302          2592  5                        END;
;  1303          2593  4                    RET_VAL = 1;
;  1304          2594  4                    EXITLOOP;
;  1305          2595  4                    END;
;  1306          2596  3                [INRANGE, OUTRANGE]:
;  1307          2597  3                    !+
;  1308          2598  3                    ! Data types which are not supported
;  1309          2599  3                    !-
;  1310          2600  3                    0;
;  1311          2601  3                TES
;  1312          2602  2            ELSE
;  1313          2603  2
;  1314          2604  3                !+
;  1315          2605  3                ! The whole rest of the buffer was scanned without finding an element separator
;  1316          2606  3                !-
;  1317          2607  3
;  1318          2608  4                BEGIN
;  1319          2609  4                LOCAL
;  1320          2610  4                    T_BUF_END;                          ! temp to hold BUF_END for deleting
;  1321          2611  4                                                        ! trailing nulls, spaces, and tabs
;  1322          2612  4                T_BUF_END = .CCB[LUB$A_BUF_END];
;  1323          2613  4
;  1324          2614  4                !+
;  1325          2615  4                ! Check the mask value and if it indicates that this string is
;  1326          2616  4                ! bound by quotes, then check to see if LUB$A_BUF_PTR is not
;  1327          2617  4                ! equal to LUB$A_BUF_END.  The assumption is that if BUF_PTR is
;  1328          2618  4                ! equal to BUF_END, then a delimiting quote was not found but
;  1329          2619  4                ! rather the SCANC stopped on end-of-record.
;  1330          2620  4                !-
;  1331          2621  4
;  1332          2622  4                IF .MASK EQL K_DBL_QUOTE OR .MASK EQL K_SGL_QUOTE
;  1333          2623  4                THEN
;  1334          2624  4                    BAS$$STOP_IO(BAS$K_DATFORERR);
```

```
;  1335          2625  4
;  1336          2626  4                    !+
;  1337          2627  4                    ! So far everything is OK.  Move the data, then check for INPUT LINE
;  1338          2628  4                    ! If this is an INPUT LINE, then we need to bump the length based on
;  1339          2629  4                    ! the terminator and move the terminator into the buffer.
;  1340          2630  4                    ! If INPUT then strip off the trailing spaces, nulls, and tabs
;  1341          2631  4                    !-
;  1342          2632  4
;  1343          2633  5                    IF (.CCB[ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_INP
;  1344          2634  5                     OR .CCB[ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_REA)
;  1345          2635  4                     AND .ELEM_TYPE EQL DSC$K_DTYPE_T
;  1346          2636  4                    THEN
;  1347          2637  4                        WHILE .(.T_BUF_END - 1)<0,8,0> EQL %C' '
;  1348          2638  4                         OR .(.T_BUF_END - 1)<0,8,0> EQL %C'	'
;  1349          2639  4                         OR .(.T_BUF_END - 1)<0,8,0> EQL %X'00'
;  1350          2640  4                        DO
;  1351          2641  4                            T_BUF_END = .T_BUF_END - 1;
;  1352          2642  4
;  1353          2643  4                    DSC[DSC$W_LENGTH] = .DSC[DSC$W_LENGTH] + (.T_BUF_END - .CCB[LUB$A_BUF_PTR]);
;  1354          2644  4                    PTRD = CH$MOVE (.T_BUF_END - .CCB[LUB$A_BUF_PTR], .PTRS, .PTRD);
;  1355          2645  4                    IF .CCB[ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_INC
;  1356          2646  4                    THEN
;  1357          2647  4
;  1358          2648  4                        !+
;  1359          2649  4                        ! This is an INPUT LINE.  Bump length and tack on the terminator
;  1360          2650  4                        !-
;  1361          2651  4
;  1362          2652  5                        BEGIN
;  1363          2653  5                        LITERAL
;  1364          2654  5                            K_ESCAPE = %X'1B',          ! ASCII escape character
;  1365          2655  5                            K_CR = %X'0D',              ! ASCII carriage return char.
;  1366          2656  5                            K_CRLF = %X'0A0D';          ! ASCII carriage return-line
;  1367          2657  5                                                        ! feed char. combination
;  1368          2658  5                    !+
;  1369          2659  5                    ! Due to an undocumented change to RMS for V2.0, we want to look only at the
;  1370          2660  5                    ! low order byte to find the terminating character.  RMS is now returning the
;  1371          2661  5                    ! length of the terminating sequence in the upper word.
;  1372          2662  5                    !-
;  1373          2663  5                        SELECTONEU .CCB [RAB$W_STV0] OF
;  1374          2664  5                        SET
;  1375          2665  5                        [K_ESCAPE]:
;  1376          2666  6                            BEGIN
;  1377          2667  6                    !+
;  1378          2668  6                    ! Check to see if the length is one.  If it is, we have to move the escape
;  1379          2669  6                    ! character by hand; it is not at the end of the buffer.  Otherwise, the escape
;  1380          2670  6                    ! sequence is at the end of the buffer following the data.
;  1381          2671  6                    !-
;  1382          2672  6                            IF .CCB [RAB$W_STV2] EQLU 1
;  1383          2673  6                            THEN
;  1384          2674  7                                BEGIN
;  1385          2675  7                                DSC [DSC$W_LENGTH] = .DSC [DSC$W_LENGTH] + 1;
;  1386          2676  7                                CH$MOVE(1, UPLIT(K_ESCAPE), .PTRD);
;  1387          2677  7                                END
;  1388          2678  6                            ELSE
;  1389          2679  7                                BEGIN
;  1390          2680  7                                DSC[DSC$W_LENGTH] = .DSC[DSC$W_LENGTH] + .CCB [RAB$W_STV2];
;  1391          2681  7                                CH$MOVE (.CCB [RAB$W_STV2], .CCB [RAB$L_RBF] + .CCB [RAB$W_RSZ], .PTRD);
```

```
;  1392          2682  6                        END;
;  1393          2683  5                    END;
;  1394          2684  5                [K_CR]:
;  1395          2685  6                    BEGIN
;  1396          2686  6                    DSC[DSC$W_LENGTH] = .DSC[DSC$W_LENGTH] + 2;
;  1397          2687  6                    CH$MOVE (2, UPLIT(K_CRLF), .PTRD);
;  1398          2688  5                    END;
;  1399          2689  5                [OTHERWISE]:
;  1400          2690  5                    ;
;  1401          2691  5                TES;
;  1402          2692  4                END;
;  1403          2693  4            CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_END];
;  1404          2694  4            RET_VAL = 1;
;  1405          2695  4            EXITLOOP;
;  1406          2696  3            END;
;  1407          2697  2        END;                ! WHILE loop
;  1408          2698  2    !+
;  1409          2699  2    ! Update the data pointer if this is a READ or MAT READ so that we are pointing
;  1410          2700  2    ! at the next data element in the event of an error.
;  1411          2701  2    !-
;  1412          2702  2        IF (.CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_MRE) OR
;  1413          2703  2         (.CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_REA)
;  1414          2704  2        THEN
;  1415          2705  3            BEGIN
;  1416          2706  3            LOCAL
;  1417          2707  3                BMF : REF BLOCK [0, BYTE] FIELD (BSF$MAJOR_FRAME);      ! BASIC major frame pointer
;  1418          2708  3            BMF = .CCB [ISB$A_MAJ_F_PTR];
;  1419          2709  3            BMF [BSF$A_CUR_DTA] = .CCB [LUB$A_BUF_PTR] + 1;
;  1420          2710  3            END;
;  1421          2711  2
;  1422          2712  2
;  1423          2713  2        !+
;  1424          2714  2        ! Convert the field that was found into internal format
;  1425          2715  2        !-
;  1426          2716  2
;  1427          2717  2        IF NOT (CASE .ELEM_TYPE
;  1428          2718  3            FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
;  1429          2719  3            SET
;  1430          2720  3            [INRANGE, OUTRANGE]:
;  1431          2721  3                !+
;  1432          2722  3                ! Data types that are not yet supported
;  1433          2723  3                !-
;  1434          2724  4                BEGIN
;  1435          2725  4                0
;  1436          2726  3                END;
;  1437          2727  3            [DSC$K_DTYPE_B]:
;  1438          2728  3                !+
;  1439          2729  3                ! Integer - byte
;  1440          2730  3                ! Do the conversion and then check the range.
;  1441          2731  3                !-
;  1442          2732  3
;  1443          2733  4                BEGIN
;  1444          2734  4                IF OTS$CVT_TI_L(DSC, ELEM[0], K_INT_SIZ, K_INT_FLAGS)
;  1445          2735  4                THEN
;  1446          2736  4
;  1447          2737  4                    !+
;  1448          2738  4                    ! The conversion was successful.
```

```
;  1449          2739  4                    !-
;  1450          2740  4
;  1451          2741  4                    IF .ELEM[0] GTR 127
;  1452          2742  4                     OR .ELEM[0] LSS -128
;  1453          2743  4                    THEN
;  1454          2744  4                        BAS$$STOP_IO (BAS$K_ILLNUM)
;  1455          2745  4                    ELSE
;  1456          2746  4                        1                       ! signify success
;  1457          2747  4                ELSE
;  1458          2748  4
;  1459          2749  4                    !+
;  1460          2750  4                    ! The conversion routine returned failure.
;  1461          2751  4                    !-
;  1462          2752  4
;  1463          2753  4                    0
;  1464          2754  3                END;
;  1465          2755  3            [DSC$K_DTYPE_W]:
;  1466          2756  3                !+
;  1467          2757  3                ! Integer - word
;  1468          2758  3                ! Do the conversion of the value input and then range check
;  1469          2759  3                ! for overflow.
;  1470          2760  3                !-
;  1471          2761  3
;  1472          2762  4                BEGIN
;  1473          2763  4                IF OTS$CVT_TI_L(DSC, ELEM[0], K_INT_SIZ, K_INT_FLAGS)
;  1474          2764  4                THEN
;  1475          2765  4
;  1476          2766  4                    !+
;  1477          2767  4                    ! The conversion was successful.  Check the range of the
;  1478          2768  4                    ! value input.  Signal an error or assume a value of success.
;  1479          2769  4                    !-
;  1480          2770  4
;  1481          2771  4                    IF .ELEM[0] GTR 32767
;  1482          2772  4                     OR .ELEM[0] LSS -32768
;  1483          2773  4                    THEN
;  1484          2774  4                        BAS$$STOP_IO (BAS$K_ILLNUM)
;  1485          2775  4                    ELSE
;  1486          2776  4                        1                   ! signify success
;  1487          2777  4                ELSE
;  1488          2778  4
;  1489          2779  4                    !+
;  1490          2780  4                    ! The conversion routine returned failure.  Assume a value of
;  1491          2781  4                    ! failure.
;  1492          2782  4                    !-
;  1493          2783  4
;  1494          2784  4                    0
;  1495          2785  3                END;
;  1496          2786  3            [DSC$K_DTYPE_L]:
;  1497          2787  3
;  1498          2788  3                !+
;  1499          2789  3                ! Integer - longword.  Upper and lower bounds checking is performed
;  1500          2790  3                ! by the conversion routine.
;  1501          2791  3                !-
;  1502          2792  3
;  1503          2793  4                BEGIN
;  1504          2794  4                OTS$CVT_TI_L(DSC, ELEM[0], K_INT_SIZ, K_INT_FLAGS)
;  1505          2795  3                END;
```

```
;  1506          2796  3              [DSC$K_DTYPE_F]:
;  1507          2797  3                  !+
;  1508          2798  3                  ! floating single precision
;  1509          2799  3                  !-
;  1510          2800  4                  BEGIN
;  1511          2801  4                  LOCAL
;  1512          2802  4                      T_ELEM: VECTOR[2];  ! temp. quadword work area
;  1513          2803  4                  IF OTS$CVT_T_D(DSC, T_ELEM, 0, 0, K_FLT_F_FLAGS)
;  1514          2804  4                      THEN LIB$CVTDF(T_ELEM[0], ELEM[0])
;  1515          2805  4                      ELSE 0
;  1516          2806  3                  END;
;  1517          2807  3              [DSC$K_DTYPE_D]:
;  1518          2808  3                  !+
;  1519          2809  3                  ! double precision floating
;  1520          2810  3                  !-
;  1521          2811  3
;  1522          2812  4                      BEGIN
;  1523          2813  4                      LOCAL
;  1524          2814  4                          STATUS;
;  1525          2815  4                      STATUS = OTS$CVT_T_D(DSC, ELEM[0], 0, .CCB [ISB$B_SCALE_FAC], K_FLT_D_FLAGS);
;  1526          2816  4 !+
;  1527          2817  4 ! Truncate any fractional portion remaining if scaling is done.
;  1528          2818  4 !-
;  1529          2819  4                      IF .CCB [ISB$B_SCALE_FAC] NEQ 0
;  1530          2820  4                      THEN
;  1531          2821  5                          BEGIN
;  1532          2822  5                          MTH$DINT(ELEM [0]);
;  1533          2823  6                              BEGIN
;  1534          2824  6                              REGISTER
;  1535          2825  6                                  R0 = 0,
;  1536          2826  6                                  R1 = 1;
;  1537          2827  6                              ELEM [0] = .R0;
;  1538          2828  6                              ELEM [1] = .R1;
;  1539          2829  5                              END;
;  1540          2830  4                      END;
;  1541          2831  4                  .STATUS
;  1542          2832  3                  END;
;  1543          2833  3              [DSC$K_DTYPE_G]:
;  1544          2834  3                  !+
;  1545          2835  3                  ! g floating
;  1546          2836  3                  !-
;  1547          2837  4                  BEGIN
;  1548          2838  4                  LOCAL
;  1549          2839  4                      STATUS;
;  1550          2840  4                  STATUS = OTS$CVT_T_G(DSC, ELEM[0], 0, 0, K_FLT_D_FLAGS);
;  1551          2841  4                  .STATUS
;  1552          2842  3                  END;
;  1553          2843  3              [DSC$K_DTYPE_H]:
;  1554          2844  3                  !+
;  1555          2845  3                  ! h floating
;  1556          2846  3                  !-
;  1557          2847  4                  BEGIN
;  1558          2848  4                  LOCAL
;  1559          2849  4                      STATUS;
;  1560          2850  4                  STATUS = OTS$CVT_T_H(DSC, ELEM[0], 0, 0, K_FLT_D_FLAGS);
;  1561          2851  4                  .STATUS
;  1562          2852  3                  END;
```

```
; 1563          2853  3              [DSC$K_DTYPE_T, DSC$K_DTYPE_P]:
; 1564          2854  3                  !+
; 1565          2855  3                  ! String or packed - no conversion - just return success
; 1566          2856  3                  !-
; 1567          2857  4                  BEGIN
; 1568          2858  4                  MAP
; 1569          2859  4                      ELEM: REF BLOCK [8, BYTE];
; 1570          2860  4                  ELEM[DSC$W_LENGTH] = .DSC[DSC$W_LENGTH];
; 1571          2861  4                  1
; 1572          2862  3                  END;
; 1573          2863  3                  TES)
; 1574          2864  2              THEN
; 1575          2865  2                  BAS$$STOP_IO(BAS$K_DATFORERR);
; 1576          2866  2          RETURN .RET_VAL;
; 1577          2867  1          END;
; INFO#250                   L1:2827
; Referenced REGISTER symbol R0 is probably not initialized
; INFO#250                   L1:2828
; Referenced REGISTER symbol R1 is probably not initialized


40 40 40 40 40 10 40 40 40 40 40 40 40 40 20  0038C P.AAC:  .BYTE    32, 64, 64, 64, 64, 64, 64, 64, 64, 16, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0039B                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
41 40 40 40 40 44 40 40 40 40 48 40 10 40 40  003AA                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
42 40 40 40 40 40 40 40 40 40 40 40 40 40 40  003B9                  64, 64, 16, 64, 72, 64, 64, 64, 64, 68, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  003C8                  64, 64, 64, 64, 65, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  003D7                  64, 64, 64, 64, 64, 64, 64, 64, 64, 66, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  003E6                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  003F5                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00404                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00413                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00422                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00431                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  00440                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0044F                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0045E                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0046D                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
40 40 40 40 40 40 40 40 40 40 40 40 40 40 40  0047C                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                          40  0048B                  64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                                                     64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                                                     64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                                                     64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                                                     64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                                                     64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                                                     64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                                                     64, 64, 64, 64, 64, 64, 64, 64, 64, 64, -   :
                                                                     64, 64, 64, 64, 64, 64                      :
                              0000001B  0048C P.AAD:  .LONG    27                                                :
                              00000A0D  00490 P.AAE:  .LONG    2573                                              :

                                                TABLE=              P.AAC

                                  07FC  00000 GETFIELD:
                                                        .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10              ; 2169
             5E                20   C2  00002           SUBL2    #32, SP                                       :
```

```
                                         18    AE  B4 00005           CLRW    DSC                                       ; 2308
                              58         B0    AB  9E 00008           MOVAB   -80(CCB), R8                              ; 2316
                                         B4    AB  9F 0000C           PUSHAB  -76(CCB)                                  ;
                        00    BE               68  D1 0000F           CMPL    (R8), @0(SP)                              ;
                                               03  1F 00013           BLSSU   1$                                        ;
                                             045F  31 00015           BRW     77$                                       ;
                                               68  D6 00018 1$:       INCL    (R8)                                      ; 2320
                                               52  D4 0001A           CLRL    R2                                        ; 2329
                        00    BE               68  D1 0001C           CMPL    (R8), @0(SP)                              ;
                                               12  12 00020           BNEQ    2$                                        ;
                                               52  D6 00022           INCL    R2                                        ;
              0B          A1  AB               04  E1 00024           BBC     #4, -95(CCB), 2$                          ; 2330
                              7E        00G    8F  9A 00029           MOVZBL  #BAS$K_TOOLITDAT, -(SP)                   ; 2332
                   00000000G  00               01  FB 0002D           CALLS   #1, BAS$$SIGNAL_IO                        ;
                              52               52  E9 00034 2$:       BLBC    R2, 9$                                    ; 2334
                              20       FF71    CB  91 00037           CMPB    -143(CCB), #32                            ; 2335
                                               4B  13 0003C           BEQL    9$                                        ;
                              50         04    AC  D0 0003E           MOVL    ELEM, R0                                  ; 2350
              16              06         08    AC  CF 00042           CASEL   ELEM_TYPE, #6, #22                        ; 2343
    002E      002E          002E             002E     00047 3$:       .WORD   4$-3$,-                                   ;
    002E      002E          0032             002E     0004F                   4$-3$,-                                   ;
    002E      002E          002E             003D     00057                   4$-3$,-                                   ;
    003D      002E          002E             002E     0005F                   4$-3$,-                                   ;
    002E      002E          002E             002E     00067                   4$-3$,-                                   ;
              0036          0032             002E     0006F                   5$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              7$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              7$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              4$-3$,-                                   ;
                                                                              5$-3$,-                                   ;
                                                                              6$-3$                                     ;
                                               60  D4 00075 4$:       CLRL    (R0)                                      ; 2357
                                               0D  11 00077           BRB     8$                                        ;
                                               60  7C 00079 5$:       CLRQ    (R0)                                      ; 2365
                                               09  11 0007B           BRB     8$                                        ; 2343
                                               60  7C 0007D 6$:       CLRQ    (R0)                                      ; 2374
                                         08    A0  7C 0007F           CLRQ    8(R0)                                     ; 2376
                                               02  11 00082           BRB     8$                                        ; 2343
                                               60  B4 00084 7$:       CLRW    (R0)                                      ; 2387
                                             0083  31 00086 8$:       BRW     18$                                       ; 2390
                        20    AE         0C    AC  D0 00089 9$:       MOVL    WORK_STR, DSC+4                           ; 2398
              16              06         08    AC  CF 0008E           CASEL   ELEM_TYPE, #6, #22                        ; 2399
    009B      0030          0030             0030     00093 10$:      .WORD   11$-10$,-                                 ;
    009B      009B          0030             0030     0009B                   11$-10$,-                                 ;
    009B      009B          009B             0036     000A3                   11$-10$,-                                 ;
    0030      009B          009B             009B     000AB                   23$-10$,-                                 ;
```

```
                009B    009B    009B    000B3                   11$-10$,-
                0030    0030    009B    000BB                   11$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                12$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                11$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                23$-10$,-
                                                                11$-10$,-
                                                                11$-10$
                        6B  11  000C1           BRB     23$
                OC  AE      31  D0  000C3 11$:  MOVL    #49, MASK                       2409
                        65  11  000C7           BRB     23$
                50  FF71    CB  9A  000C9 12$:  MOVZBL  -143(CCB), R0                   2423
                1C          50  91  000CE       CMPB    R0, #28
                            0A  13  000D1       BEQL    13$
                20          50  91  000D3       CMPB    R0, #32                         2424
                            05  13  000D6       BEQL    13$
                32          50  91  000D8       CMPB    R0, #50                         2425
                            05  12  000DB       BNEQ    14$
                    OC  AE  D4  000DD 13$:  CLRL    MASK                            2427
                        4C  11  000E0       BRB     23$
                20  00      B8  91  000E2 14$:  CMPB    @0(R8), #32                     2436
                            0B  13  000E6       BEQL    15$
                09  00      B8  91  000E8       CMPB    @0(R8), #9                      2437
                            05  13  000EC       BEQL    15$
                    00      B8  95  000EE       TSTB    @0(R8)                          2438
                            0A  12  000F1       BNEQ    16$
            00  BE          68  D1  000F3 15$:  CMPL    (R8), @0(SP)                    2439
                            04  18  000F7       BGEQ    16$
                            68  D6  000F9       INCL    (R8)                            2441
                            E5  11  000FB       BRB     14$
            00  BE          68  D1  000FD 16$:  CMPL    (R8), @0(SP)                    2442
                            06  18  00101       BGEQ    17$
                2C  00      B8  91  00103       CMPB    @0(R8), #44                     2443
                            07  12  00107       BNEQ    19$
                    04      BC  B4  00109 17$:  CLRW    @ELEM                           2448
                50          01  D0  0010C 18$:  MOVL    #1, R0                          2449
                            04  0010F          RET
                27  00      B8  91  00110 19$:  CMPB    @0(R8), #39                     2451
                            06  12  00114       BNEQ    20$
            0C  AE          04  D0  00116       MOVL    #4, MASK                        2454
                            0A  11  0011A       BRB     21$                             2455
                22  00      B8  91  0011C 20$:  CMPB    @0(R8), #34                     2458
                            08  12  00120       BNEQ    22$
            0C  AE          08  D0  00122       MOVL    #8, MASK                        2461
                            68  D6  00126 21$:  INCL    (R8)                            2462
                            04  11  00128       BRB     23$                             2458
```

```
                        0C  AE              01  D0 0012A 22$:    MOVL    #1, MASK                                  ; 2465
                            5A              68  D0 0012E 23$:    MOVL    (R8), PTRS                                ; 2473
                        04  AE      20      AE  D0 00131         MOVL    DSC+4, PTRD                               ; 2474
                 57     00  BE              68  C3 00136         SUBL3   (R8), @0(SP), LEN                         ; 2475
                  00010000  8F              57  D1 0013B 24$:    CMPL    LEN, #65536                               ; 2487
                                            07  1F 00142         BLSSU   25$                                       ;
                            50    FFFF      8F  3C 00144         MOVZWL  #65535, TEMP_LEN                          ;
                                            03  11 00149         BRB     26$                                       ;
                            50              57  D0 0014B 25$:    MOVL    LEN, TEMP_LEN                             ;
   0C    AE    FDA3  CF     00  B8          50  2A 0014E 26$:    SCANC   TEMP_LEN, @0(R8), TABLE, MASK             ; 2488
                                            02  12 00157         BNEQ    27$                                       ;
                                            51  D4 00159         CLRL    R1                                        ;
                            56              51  D0 0015B 27$:    MOVL    R1, SCAN_VAL                              ;
                                            03  12 0015E         BNEQ    28$                                       ; 2489
                                          0117  31 00160         BRW     45$                                       ;
                 16         06      08      AC  CF 00163 28$:    CASEL   ELEM_TYPE, #6, #22                        ; 2491
      FFD3     0030       0030            0030     00168 29$:    .WORD   31$-29$,-
      FFD3     FFD3       0030            0030     00170                 31$-29$,-
      FFD3     FFD3       FFD3            0081     00178                 31$-29$,-
      0030     FFD3       FFD3            FFD3     00180                 24$-29$,-
      FFD3     FFD3       FFD3            FFD3     00188                 31$-29$,-
               0030       0030            FFD3     00190                 31$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         35$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         31$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         24$-29$,-
                                                                         31$-29$,-
                                                                         31$-29$
                                            A3  11 00196 30$:    BRB     24$                                       ;
                 59         56              68  C3 00198 31$:    SUBL3   (R8), SCAN_VAL, R9                        ; 2497
            04   BE         6A              59  28 0019C         MOVC3   R9, (PTRS), @PTRD                         ;
                            09              66  91 001A1         CMPB    (SCAN_VAL), #9                            ; 2498
                                            09  13 001A4         BEQL    32$                                       ;
                            20              66  91 001A6         CMPB    (SCAN_VAL), #32                           ; 2499
                                            04  13 001A9         BEQL    32$                                       ;
                                            66  95 001AB         TSTB    (SCAN_VAL)                                ; 2500
                                            1B  12 001AD         BNEQ    33$                                       ;
                        1C  AE              59  A0 001AF 32$:    ADDW2   R9, DSC                                   ; 2510
                            5A      01    A94A  9E 001B3         MOVAB   1(R9)[PTRS], PTRS                         ; 2511
                        04  AE              59  C0 001B8         ADDL2   R9, PTRD                                  ; 2512
                 50         57              59  C3 001BC         SUBL3   R9, LEN, R0                               ; 2513
                            57      FF      A0  9E 001C0         MOVAB   -1(R0), LEN                               ;
                            68      01      A6  9E 001C4         MOVAB   1(R6), (R8)                               ; 2514
                                            CC  11 001C8         BRB     30$                                       ; 2498
                            68              56  D1 001CA 33$:    CMPL    SCAN_VAL, (R8)                            ; 2518
                                            17  13 001CD         BEQL    34$                                       ;
```

```
                                50    1C  AE  3C 001CF        MOVZWL  DSC, R0                                  ; 2529
                                50        56  C0 001D3        ADDL2   SCAN_VAL, R0                             ;
              1C  AE            50        68  A3 001D6        SUBW3   (R8), R0, DSC                            ;
                  50            57        59  C3 001DB        SUBL3   R9, LEN, R0                              ; 2530
                                57    FF  A0  9E 001DF        MOVAB   -1(R0), LEN                              ;
                                68        56  D0 001E3        MOVL    SCAN_VAL, (R8)                           ; 2531
                                        012D  31 001E6 34$:   BRW     55$                                      ; 2532
                        08      AE        56  D0 001E9 35$:   MOVL    SCAN_VAL, A_HIGH_MARK                    ; 2552
                                01    0C  AE  D1 001ED        CMPL    MASK, #1                                 ; 2553
                                          15  12 001F1        BNEQ    38$                                      ;
                                20    FF  A6  91 001F3 36$:   CMPB    -1(SCAN_VAL), #32                        ; 2555
                                          0B  13 001F7        BEQL    37$                                      ;
                                09    FF  A6  91 001F9        CMPB    -1(SCAN_VAL), #9                         ; 2556
                                          05  13 001FD        BEQL    37$                                      ;
                                      FF  A6  95 001FF        TSTB    -1(SCAN_VAL)                             ; 2557
                                          04  12 00202        BNEQ    38$                                      ;
                                          56  D7 00204 37$:   DECL    SCAN_VAL                                 ; 2559
                                          EB  11 00206        BRB     36$                                      ;
                  59            56        68  C3 00208 38$:   SUBL3   (R8), SCAN_VAL, R9                       ; 2561
                        1C      AE        59  B0 0020C        MOVW    R9, DSC                                  ;
              04  BE            6A        59  28 00210        MOVC3   R9, (PTRS), @PTRD                        ; 2562
                                68    08  AE  D0 00215        MOVL    A_HIGH_MARK, (R8)                        ; 2568
                                27    08  BE  91 00219        CMPB    @A_HIGH_MARK, #39                        ; 2569
                                          06  13 0021D        BEQL    39$                                      ;
                                22    08  BE  91 0021F        CMPB    @A_HIGH_MARK, #34                        ;
                                          52  12 00223        BNEQ    44$                                      ;
                                          68  D6 00225 39$:   INCL    (R8)                                     ; 2576
                        0C      AE    41  8F  9A 00227        MOVZBL  #65, MASK                                ; 2583
                                50    1C  AE  3C 0022C        MOVZWL  DSC, R0                                  ; 2584
                  50            57        50  C3 00230        SUBL3   R0, LEN, R0                              ;
                                          50  D7 00234        DECL    REM_LENGTH                               ;
                  00010000      8F        50  D1 00236        CMPL    REM_LENGTH, #65536                       ; 2585
                                          05  1F 0023D        BLSSU   40$                                      ;
                                50  FFFF  8F  3C 0023F        MOVZWL  #65535, REM_LENGTH                       ;
  0C   AE   FCAD  CF        00  B8        50  2A 00244 40$:   SCANC   REM_LENGTH, @0(R8), TABLE, MASK          ; 2586
                                          02  12 0024D        BNEQ    41$                                      ;
                                          51  D4 0024F        CLRL    R1                                       ;
                                50        51  D0 00251 41$:   MOVL    R1, T_RET_VAL                            ;
                                          07  12 00254        BNEQ    42$                                      ; 2588
                  51        00  BE        01  C1 00256        ADDL3   #1, @0(SP), R1                           ;
                                          03  11 0025B        BRB     43$                                      ;
                                51        50  D0 0025D 42$:   MOVL    T_RET_VAL, R1                            ;
                                68        51  D0 00260 43$:   MOVL    R1, (R8)                                 ;
                                          50  D5 00263        TSTL    T_RET_VAL                                ; 2589
                                          10  13 00265        BEQL    44$                                      ;
                                2C        60  91 00267        CMPB    (T_RET_VAL), #44                         ; 2590
                                          0B  13 0026A        BEQL    44$                                      ;
                                7E   00G  8F  9A 0026C        MOVZBL  #BAS$K_DATFORERR, -(SP)                  ; 2591
                     00000000G  00        01  FB 00270        CALLS   #1, BAS$$STOP_IO                         ;
                                        009C  31 00277 44$:   BRW     55$                                      ; 2593
                                52    00  BE  D0 0027A 45$:   MOVL    @0(SP), T_BUF_END                        ; 2612
                                08    0C  AE  D1 0027E        CMPL    MASK, #8                                 ; 2622
                                          06  13 00282        BEQL    46$                                      ;
                                04    0C  AE  D1 00284        CMPL    MASK, #4                                 ;
                                          0B  12 00288        BNEQ    47$                                      ;
                                7E   00G  8F  9A 0028A 46$:   MOVZBL  #BAS$K_DATFORERR, -(SP)                  ; 2624
                     00000000G  00        01  FB 0028E        CALLS   #1, BAS$$STOP_IO                         ;
```

```
                        59      FF71    CB  9A 00295 47$:     MOVZBL  -143(CCB), R9                   ; 2633
                        1E              59  91 0029A           CMPB    R9, #30
                                        05  13 0029D           BEQL    48$
                        22              59  91 0029F           CMPB    R9, #34                         ; 2634
                                        1B  12 002A2           BNEQ    51$
                        0E      08      AC  D1 002A4 48$:      CMPL    ELEM_TYPE, #14                  ; 2635
                                        15  12 002A8           BNEQ    51$
                        20      FF      A2  91 002AA 49$:      CMPB    -1(T_BUF_END), #32              ; 2637
                                        0B  13 002AE           BEQL    50$
                        09      FF      A2  91 002B0           CMPB    -1(T_BUF_END), #9               ; 2638
                                        05  13 002B4           BEQL    50$
                                FF      A2  95 002B6           TSTB    -1(T_BUF_END)                   ; 2639
                                        04  12 002B9           BNEQ    51$
                                        52  D7 002BB 50$:      DECL    T_BUF_END                       ; 2641
                                        EB  11 002BD           BRB     49$
                        52              68  C2 002BF 51$:      SUBL2   (R8), R2                        ; 2643
                1C      AE              52  A0 002C2           ADDW2   R2, DSC
04      BE      6A                      52  28 002C6           MOVC3   R2, (PTRS), @PTRD               ; 2644
                04      AE              53  D0 002CB           MOVL    R3, PTRD
                        20              59  91 002CF           CMPB    R9, #32                         ; 2645
                                        3E  12 002D2           BNEQ    54$
                        50      0C      AB  3C 002D4           MOVZWL  12(CCB), R0                     ; 2663
                        1B              50  B1 002D8           CMPW    R0, #27                         ; 2665
                                        26  12 002DB           BNEQ    53$
                        01      0E      AB  B1 002DD           CMPW    14(CCB), #1                     ; 2672
                                        0B  12 002E1           BNEQ    52$
                                1C      AE  B6 002E3           INCW    DSC                             ; 2675
                04      BE      FD0E    CF  90 002E6           MOVB    P.AAD, @PTRD                    ; 2676
                                        24  11 002EC           BRB     54$                             ; 2672
                1C      AE      0E      AB  A0 002EE 52$:      ADDW2   14(CCB), DSC                    ; 2680
                        50      22      AB  3C 002F3           MOVZWL  34(CCB), R0                     ; 2681
                        50      28      AB  C0 002F7           ADDL2   40(CCB), R0
        04      BE              60      0E  AB 28 002FB        MOVC3   14(CCB), (R0), @PTRD
                                        0F  11 00301           BRB     54$
                        0D              50  B1 00303 53$:      CMPW    R0, #13                         ; 2663
                                        0A  12 00306           BNEQ    54$                             ; 2684
                1C      AE              02  A0 00308           ADDW2   #2, DSC                         ; 2686
                04      BE      FCEC    CF  B0 0030C           MOVW    P.AAE, @PTRD                    ; 2687
                        68      00      BE  D0 00312 54$:      MOVL    @0(SP), (R8)                    ; 2693
                10      AE              01  D0 00316 55$:      MOVL    #1, RET_VAL                     ; 2694
                        36      FF71    CB  91 0031A           CMPB    -143(CCB), #54                  ; 2702
                                        07  13 0031F           BEQL    56$
                        22      FF71    CB  91 00321           CMPB    -143(CCB), #34                  ; 2703
                                        0B  12 00326           BNEQ    57$
                        50      FF4B    CB  D0 00328 56$:      MOVL    -184(CCB), BMF                  ; 2708
        0087    C0              68      01  C1 0032D           ADDL3   #1, (R8), 135(BMF)              ; 2709
                16              06      08  AC CF 00333 57$:   CASEL   ELEM_TYPE, #6, #22              ; 2717
012F            008E            0059            0031    00338 58$:  .WORD   60$-58$,-
012F            012F            00C6            00A1    00340               61$-58$,-
012F            012F            012F            0128    00348               64$-58$,-
0128            012F            012F            012F    00350               75$-58$,-
012F            012F            012F            012F    00358               66$-58$,-
                0110            00FB            012F    00360               67$-58$,-
                                                                            75$-58$,-
                                                                            75$-58$,-
                                                                            74$-58$,-
                                                                            75$-58$,-
```

```
                                                               75$-58$,-                              :
                                                               75$-58$,-                              :
                                                               75$-58$,-                              :
                                                               75$-58$,-                              :
                                                               75$-58$,-                              :
                                                               74$-58$,-                              :
                                                               75$-58$,-                              :
                                                               75$-58$,-                              :
                                                               75$-58$,-                              :
                                                               75$-58$,-                              :
                                                               75$-58$,-                              :
                                                               70$-58$,-                              :
                                                               72$-58$                                :
                          00FE   31 00366 59$:      BRW        75$                                    : 2724
                            05   DD 00369 60$:      PUSHL      #5                                     : 2734
                            04   DD 0036B           PUSHL      #4                                     :
                     04     AC   DD 0036D           PUSHL      ELEM                                   :
                     28     AE   9F 00370           PUSHAB     DSC                                    :
0000000G       00           04   FB 00373           CALLS      #4, OTS$CVT_TI_L                       :
               E9           50   E9 0037A           BLBC       R0, 59$                                :
0000007F       8F    04     BC   D1 0037D           CMPL       @ELEM, #127                            : 2741
                            32   14 00385           BGTR       63$                                    :
FFFFFF80       8F    04     BC   D1 00387           CMPL       @ELEM, #-128                           : 2742
                            26   11 0038F           BRB        62$                                    :
                            05   DD 00391 61$:      PUSHL      #5                                     : 2763
                            04   DD 00393           PUSHL      #4                                     :
                     04     AC   DD 00395           PUSHL      ELEM                                   :
                     28     AE   9F 00398           PUSHAB     DSC                                    :
0000000G       00           04   FB 0039B           CALLS      #4, OTS$CVT_TI_L                       :
               C1           50   E9 003A2           BLBC       R0, 59$                                :
00007FFF       8F    04     BC   D1 003A5           CMPL       @ELEM, #32767                          : 2771
                            0A   14 003AD           BGTR       63$                                    :
FFFF8000       8F    04     BC   D1 003AF           CMPL       @ELEM, #-32768                         : 2772
                            78   18 003B7 62$:      BGEQ       69$                                    :
               7E         00G    8F   9A 003B9 63$: MOVZBL     #BAS$K_ILLNUM, -(SP)                   : 2774
0000000G       00           01   FB 003BD           CALLS      #1, BAS$$STOP_IO                       :
                            11   11 003C4           BRB        65$                                    :
                            05   DD 003C6 64$:      PUSHL      #5                                     : 2794
                            04   DD 003C8           PUSHL      #4                                     :
                     04     AC   DD 003CA           PUSHL      ELEM                                   :
                     28     AE   9F 003CD           PUSHAB     DSC                                    :
0000000G       00           04   FB 003D0           CALLS      #4, OTS$CVT_TI_L                       :
                            6D   11 003D7 65$:      BRB        71$                                    :
               7E    7B     8F   9A 003D9 66$:      MOVZBL     #123, -(SP)                            : 2803
                            7E   7C 003DD           CLRQ       -(SP)                                  :
                     20     AE   9F 003DF           PUSHAB     T_ELEM                                 :
                     2C     AE   9F 003E2           PUSHAB     DSC                                    :
0000000G       00           05   FB 003E5           CALLS      #5, OTS$CVT_T_D                        :
               78           50   E9 003EC           BLBC       R0, 75$                                :
                     04     AC   DD 003EF           PUSHL      ELEM                                   : 2804
                     18     AE   9F 003F2           PUSHAB     T_ELEM                                 :
0000000G       00           02   FB 003F5           CALLS      #2, LIB$CVTDF                          :
                            5D   11 003FC           BRB        73$                                    :
               7E    73     8F   9A 003FE 67$:      MOVZBL     #115, -(SP)                            : 2815
               7E  FF70     CB   98 00402           CVTBL      -144(CCB), -(SP)                       :
                            7E   D4 00407           CLRL       -(SP)                                  :
               52    04     AC   D0 00409           MOVL       ELEM, R2                               :
```

```
                                52  DD 0040D            PUSHL   R2                                                          ;
                             2C AE  9F 0040F            PUSHAB  DSC                                                         ;
                00000000G    00     05 FB 00412         CALLS   #5, OTS$CVT_T_D                                             ;
                             53     50 D0 00419         MOVL    R0, STATUS                                                  ;
                           FF70     CB 95 0041C         TSTB    -144(CCB)                                                   ; 2819
                                    0C 13 00420         BEQL    68$                                                         ;
                                    52 DD 00422         PUSHL   R2                                                          ; 2822
                00000000G    00     01 FB 00424         CALLS   #1, MTH$DINT                                                ;
                             62     50 7D 00428         MOVQ    R0, (R2)                                                    ; 2827
                             36     53 E9 0042E 68$:    BLBC    STATUS, 75$                                                 ; 2831
                                    3F 11 00431 69$:    BRB     76$                                                         ;
                             7E   73 8F 9A 00433 70$:    MOVZBL  #115, -(SP)                                                 ; 2840
                                    7E 7C 00437         CLRQ    -(SP)                                                       ;
                             04     AC DD 00439         PUSHL   ELEM                                                        ;
                             2C     AE 9F 0043C         PUSHAB  DSC                                                         ;
                00000000G    00     05 FB 0043F         CALLS   #5, OTS$CVT_T_G                                             ;
                                    13 11 00446 71$:    BRB     73$                                                         ; 2841
                             7E   73 8F 9A 00448 72$:    MOVZBL  #115, -(SP)                                                 ; 2850
                                    7E 7C 0044C         CLRQ    -(SP)                                                       ;
                             04     AC DD 0044E         PUSHL   ELEM                                                        ;
                             2C     AE 9F 00451         PUSHAB  DSC                                                         ;
                00000000G    00     05 FB 00454         CALLS   #5, OTS$CVT_T_H                                             ;
                             09     50 E9 0045B 73$:    BLBC    STATUS, 75$                                                 ; 2851
                                    12 11 0045E         BRB     76$                                                         ;
                          04 BC   1C AE B0 00460 74$:    MOVW    DSC, @ELEM                                                  ; 2860
                                    0B 11 00465         BRB     76$                                                         ;
                             7E  00G 8F 9A 00467 75$:    MOVZBL  #BAS$K_DATFORERR, -(SP)                                     ; 2865
                00000000G    00     01 FB 0046B         CALLS   #1, BAS$$STOP_IO                                            ;
                             50   10 AE D0 00472 76$:    MOVL    RET_VAL, R0                                                 ; 2866
                                    04 00476            RET                                                                 ;
                                50  D4 00477 77$:    CLRL    R0                                                          ; 2867
                                    04 00479            RET                                                                 ;

; Routine Size:  1146 bytes,    Routine Base:  _BAS$CODE + 0494

; 1578          2868  1 ·    END
; 1579          2869  0 ELUDOM
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _BAS$CODE                       2318  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                                 Library Statistics
;
;                                          -------- Symbols --------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped      Time
;
```

```
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1          9776        22         0        581          00:01.2



; Information:  2
; Warnings:     0
; Errors:       0


;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:BASUDFRL/OBJ=OBJ$:BASUDFRL MSRC$:BASUDFRL/UPDATE=(ENH$:BASUDFRL)

; 1580           2870  0
; Size:          2043 code + 275 data bytes
; Run Time:         00:45.4
; Elapsed Time:     01:40.2
; Lines/CPU Min:     3797
; Lexemes/CPU-Min: 25373
; Memory Used:  428 pages
; Compilation Complete
```

BASUDFWF
LIS

BASSTRING
LIS

BASTERMIO
LIS

BASTRM
LIS

BASTAB
LIS

BASUDFRM
LIS

BASSYS
LIS

BASSTR
LIS

BASUDFR
LIS